PISO-DNM100-D/T
DeviceNet Master PCI Board
User's Manual

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 2007 by ICP DAS Co., LTD. All rights reserved worldwide.

Trademark

The names used for identification only may be registered trademarks of their respective companies.

Contents

1. GENERAL INFORMATION 5

1.1 DEVICENET INTRODUCTION 5
1.2 DEVICENET APPLICATIONS 7
1.3 PISO-DNM100 ARCHITECTURE 8
1.4 DEVICENET MASTER CHARACTERISTICS 9
1.5 PISO-DNM100 FIRMWARE CHARACTERISTICS 12
1.6 FEATURES 14
1.7 SPECIFICATIONS 15
1.8 BLOCK DIAGRAM 16
1.9 PRODUCT CHECK LIST 17

2. HARDWARE CONFIGURATION 18

2.1 BOARD LAYOUT 18
2.2 JUMPER SELECTION 19
2.3 CONNECTOR PIN ASSIGNMENT 20
2.3.1 5-pin screw terminal connector 20
2.3.2 9-pin D-sub male connector 21
2.3.3 Wire connection 22
2.4 INDICATOR LED 23
2.4.1 Green LED 23
2.4.2 Red LED 23
2.5 HARDWARE INSTALLATION 24

3. DRIVER INSTALLATION AND SOFTWARE APPLICATION 25

3.1 DRIVER INSTALLATION OF THE PISO-DNM100 26
3.2 FLOW DIAGRAM FOR DEBUGGING 31
3.3 FLOW DIAGRAM FOR I/O CONFIGURATION 32
3.4 FLOW DIAGRAM FOR GENERAL I/O OPERATION 33
3.5 FLOW DIAGRAM FOR ON-LINE ADDING/REMOVING DEVICE 34
3.6 FLOW DIAGRAM FOR "SETATTRIBUTE" AND "GETATTRIBUTE" 36
3.7 FLOW DIAGRAM FOR POLL CONNECTION 37
3.8 FLOW DIAGRAM FOR BIT-STROBE CONNECTION 38
3.9 FLOW DIAGRAM FOR COS CONNECTION 39
3.10 FLOW DIAGRAM FOR CYCLIC CONNECTION 40

4. FUNCTION DESCRIPTION 41

4.1 DLL FUNCTION DEFINITION AND DESCRIPTION 42

4.2 FUNCTION RETURN CODE 48
4.3 FLOW DIAGRAM FOR AUTO-CHECK UCMM ARCHITECTURE 61
4.4 FUNCTION DESCRIPTION 62
4.4.1 DNM100_GetBoardInf 62
4.4.2 DNM100_TotalDNM100Board 63
4.4.3 DNM100_ActiveBoard 64
4.4.4 DNM100_CloseBoard 65
4.4.5 DNM100_GetDLLVersion 66
4.4.6 DNM100_GetFirmwareVersion 67
4.4.7 DNM100_ResetFirmware 68
4.4.8 DNM100_CheckFirmwareMode 69
4.4.9 DNM100_GetMasterMACID 70
4.4.10 DNM100_SetMasterMACID 71
4.4.11 DNM100_GetBaudRate 72
4.4.12 DNM100_SetBaudRate 73
4.4.13 DNM100_GetMasterStatus 74
4.4.14 DNM100_StartDevice 75
4.4.15 DNM100_StopDevice 76
4.4.16 DNM100_StartAllDevice 77
4.4.17 DNM100_StopAllDevice 78
4.4.18 DNM100_AddDevice 79
4.4.19 DNM100_RemoveDevice 80
4.4.20 DNM100_GetExplicitStatus 81
4.4.21 DNM100_GetExplicitResult 82
4.4.22 DNM100_GetAttribute 83
4.4.23 DNM100_GetAttributeValue 84
4.4.24 DNM100_SetAttribute 85
4.4.25 DNM100_SetAttributeResponse 86
4.4.26 DNM100_RemoveIOConnection 87
4.4.27 DNM100_SetBreakOption 88
4.4.28 DNM100_GetBreakOption 89
4.4.29 DNM100_ClearAllConfig 90
4.4.30 DNM100_GetLastDeviceError 91
4.4.31 DNM100_GetDeviceStatus 92
4.4.32 DNM100_DebugDevice 93
4.4.33 DNM100_GetDeviceInfo 94
4.4.34 DNM100_GetDevicePollInfo 95
4.4.35 DNM100_GetDeviceBitStrobeInfo 96

4.4.36 DNM100_GetDeviceCOSInfo 97
4.4.37 DNM100_GetDeviceCyclicInfo 98
4.4.38 DNM100_AutoScanDevice 99
4.4.39 DNM100_GetScanList 100
4.4.40 DNM100_ConfigPoll 101
4.4.41 DNM100_UpdatePollConfig 102
4.4.42 DNM100_ReadPollInputData 103
4.4.43 DNM100_WritePollOutputData 104
4.4.44 DNM100_GetPollStatus 105
4.4.45 DNM100_GetPollResult 106
4.4.46 DNM100_CheckPollConnectionStatus 107
4.4.47 DNM100_ConfigBitStrobe 108
4.4.48 DNM100_UpdateBitStrobeConfig 109
4.4.49 DNM100_ReadBitStrobeInputData 110
4.4.50 DNM100_GetBitStrobeStatus 111
4.4.51 DNM100_GetBitStrobeResult 112
4.4.52 DNM100_CheckBitStrobeConnectionStatus 113
4.4.53 DNM100_ConfigCOS 114
4.4.54 DNM100_UpdateCOSConfig 115
4.4.55 DNM100_ReadCOSInputData 116
4.4.56 DNM100_WriteCOSOutputData 117
4.4.57 DNM100_GetCOSStatus 118
4.4.58 DNM100_GetCOSResult 119
4.4.59 DNM100_CheckCOSConnectionStatus 120
4.4.60 DNM100_ConfigCyclic 121
4.4.61 DNM100_UpdateCyclicConfig 122
4.4.62 DNM100_ReadCyclicInputData 123
4.4.63 DNM100_WriteCyclicOutputData 124
4.4.64 DNM100_GetCyclicStatus 125
4.4.65 DNM100_GetCyclicResult 126
4.4.66 DNM100_CheckCyclicConnectionStatus 127

5. DEMO PROGRAMS FOR WINDOWS 128

5.1 A BRIEF INTRODUCTION TO THE DEMO PROGRAMS 128
5.2 WIRE CONNECTION OF THE CAN BUS 129
5.3 VC_DEMO1 INTRODUCTION 130
5.4 VC_DEMO2 INTRODUCTION 133
5.5 BCB_DEMO1 INTRODUCTION 135
5.6 BCB_DEMO2 INTRODUCTION 137

1. General Information

1.1 DeviceNet Introduction

The CAN (Controller Area Network) is a serial communication protocol, which efficiently supports distributed real-time control with a very high level of security. It is an especially suited for networking "intelligent" devices as well as sensors and actuators within a system or sub-system. In CAN networks, there is no addressing of subscribers or stations in the conventional sense, but instead, prioritized messages are transmitted. DeviceNet is one kind of the network protocols based on the CAN bus and mainly used for machine control network, such as textile machinery, printing machines, injection molding machinery, or packaging machines, etc. DeviceNet is a low level network that provides connections between simple industrial devices (sensors, actuators) and higher-level devices (controllers), as shown in Figure 1.1.

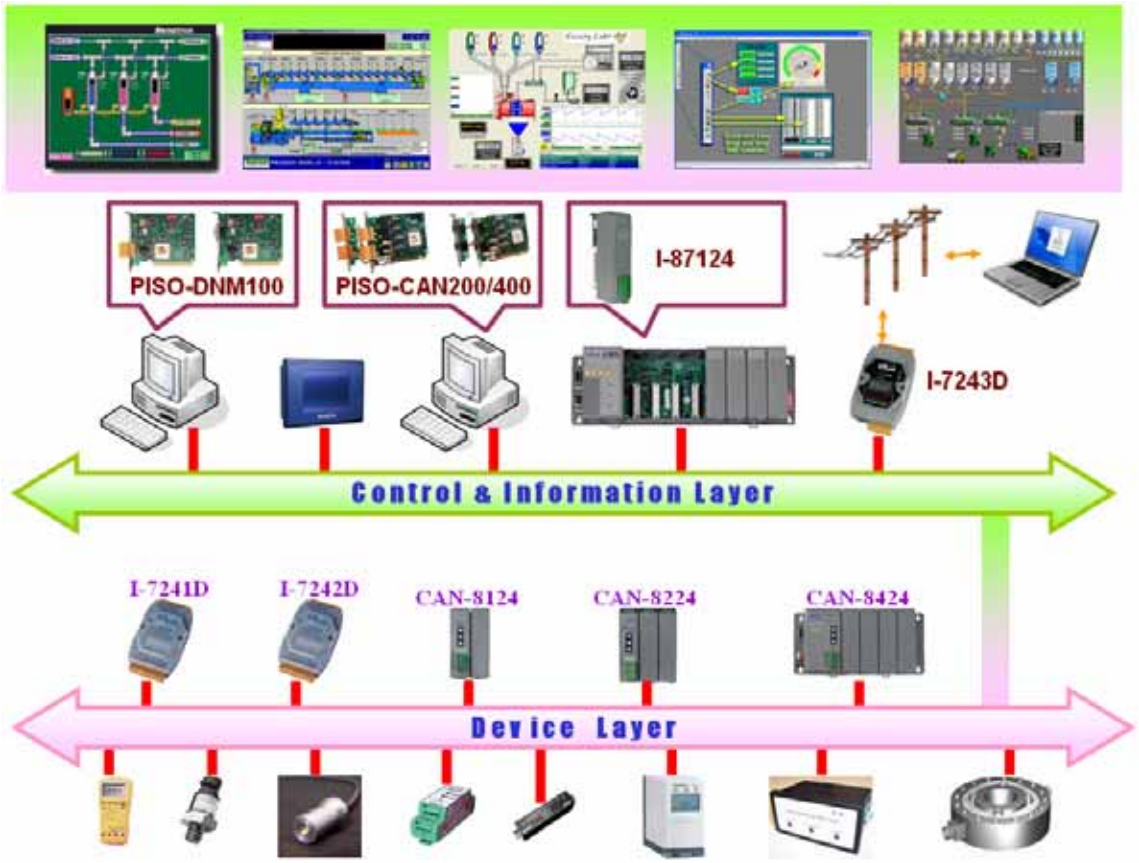


Figure 1.1 Example of the DeviceNet network

DeviceNet is a cost effective solution to one kind application of control c\area network. It reduces the connection wires between devices and provides rapid troubleshooting rejection function. The transfer rate can be up to 500Kbps within 100 meters. The transfer distance can be up to 500 meters in 125Kbps (See Table 1.1). It allows direct peer to peer data exchange between nodes in an organized and, if necessary, deterministic manner. Master/Slave connection model can be supported in the same network. Therefore, DeviceNet is able to facilitate all application communications based on a redefine a connection scheme. However, DeviceNet connection object strands as the communication path between multiple endpoints, which are application objects that is needed to share data.

Baud rate (bit/s)	Max. Bus length (m)
500 K	100
250 K	250
125 K	500

Table 1.1 The Baud rate and the Bus length

PISO-DNM100 can represent an economic solution of DeviceNet application and be a DeviceNet master device on the DeviceNet network. PISO-DNM100 supports Group 2 only Server and UCMM functions to communication with slave devices. It has an independent CAN bus communication port with the ability to cover a wide range of DeviceNet applications. Besides, PISO-DNM100 uses the new CAN controller Phillips SJA1000T and transceiver 82C250/251, which provide bus arbitration, error detection with auto correction and re-transmission function. It can be installed on almost any windows-based system, for example Win98/Win2000/WinXP/WinNT. It is popularly applied in the industrial automation, building automation, vehicle, marine, and embedded control network. Therefore, that is an easy way to develop the DeviceNet network with PISO-DNM100.

1.2 DeviceNet Applications

DeviceNet is the standardized network application layer optimized for factory automation. It is mainly used in low- and mid-volume automation systems. Some users have also implemented DeviceNet for machine control systems. The main DeviceNet application fields include the following application area (For more information, please refer to www.odva.org):

- Production cell builds and tests CPUs
- Beer brewery
- Equipment for food packing
- Fiberglass twist machine
- Sponge production plant
- Isolation wall manufacturing
- Overhead storage bin production
- Pocket-bread bakery
- Dinnerware production
- HVAC module production
- Textile machines
- Trawler automation system
- LCD manufacturing plant
- Rolling steel door production
- Bottling line
- Tight manufacturing

1.3 PISO-DNM100 Architecture

The PISO-DNM100 provides users to establish DeviceNet network rapidly by Master/Slave connection model. The PISO-DNM100 is a high-performance DeviceNet master board with one CPU inside. This architecture of the PISO-DNM100 almost doesn't cost CPU resource and really increases the work efficiency on DeviceNet network. Applying the PISO-DNM100, users don't need to take care of the detail of the DeviceNet protocol. The inside firmware implements the DeviceNet protocol to help users to establish the connection with DeviceNet slave devices easily. The illustration about the idea is shown as Figure 1.2.

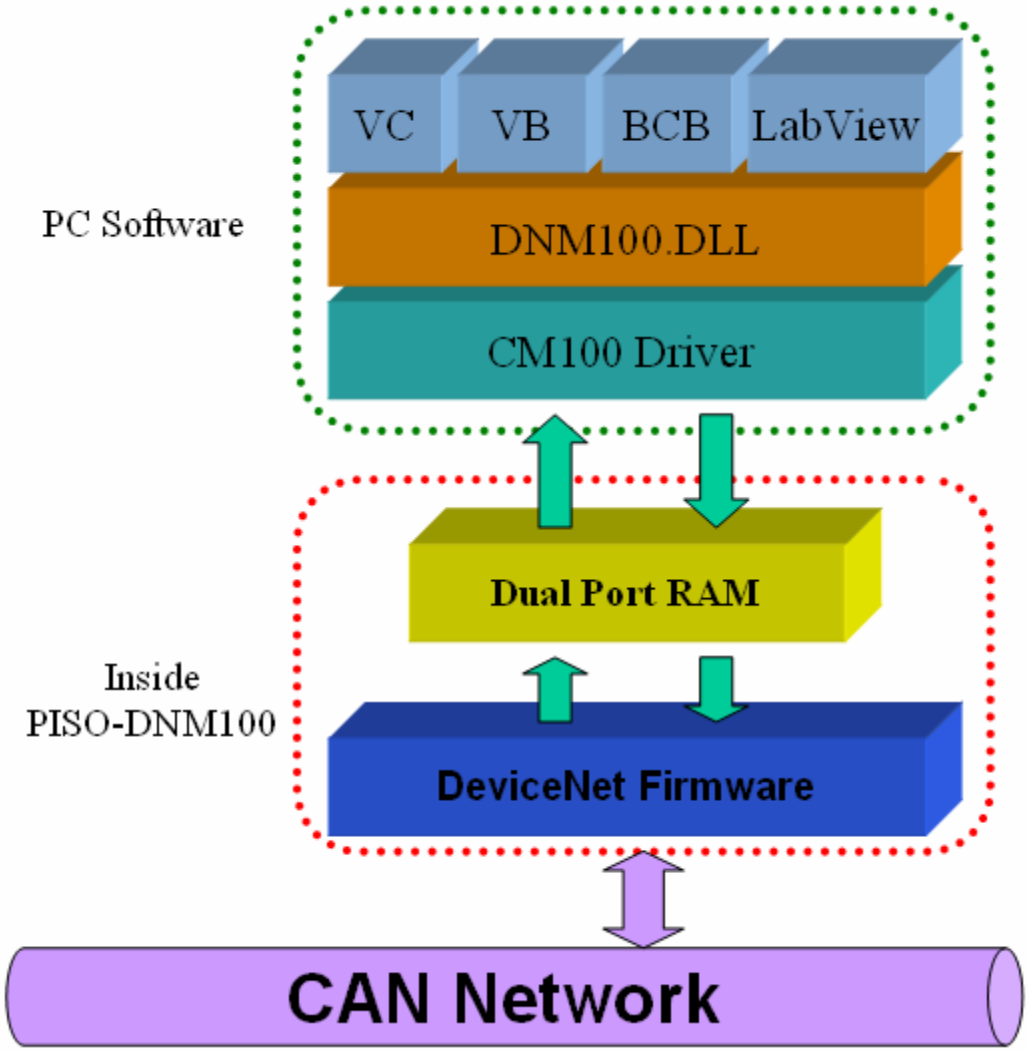


Figure 1.2 PISO-DNM100 illustration.

1.4 DeviceNet Master Characteristics

Using the API functions, users don't need to take care of the detail of the DeviceNet protocol. It can reduce the complexity of user's DeviceNet Master Software. The firmware mainly supports the Predefined Master-Slave Connection Set and UCMM functions to allow users to merge third party's DeviceNet devices into the DeviceNet network. It can help users to establish the connection with DeviceNet slave devices easily. The general application architecture is demonstrated as Figure 1.3.

Figure 1.3 Application architecture

The DeviceNet protocol firmware provides the DeviceNet Master mechanism to communicate with slave devices by the Predefined Master/Slave Connection Set and UCMM Connection Set. In the DeviceNet communication protocol can be clarify as two forms: One is the Explicit Message and others are I/O Messages. Here, we only provide one explicit message connection and four I/O connections as depicted in Figure 1.4.

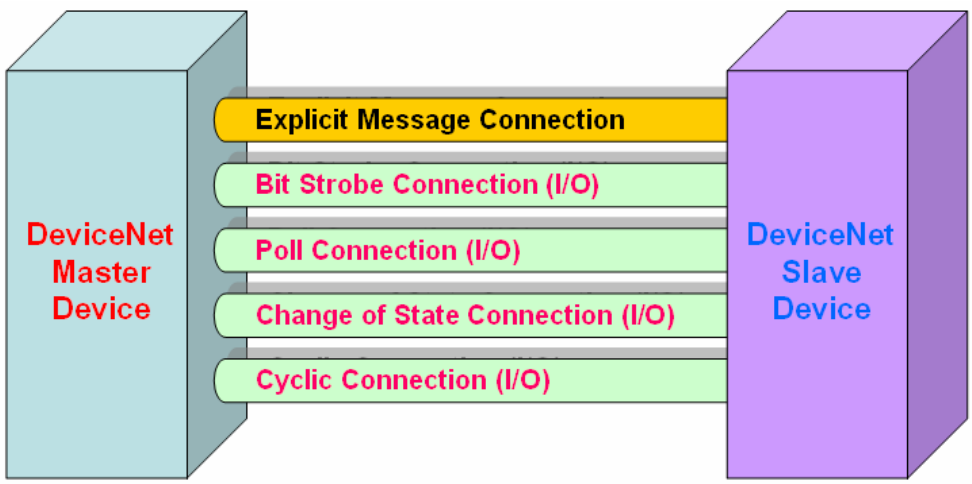


Figure 1.4 DeviceNet Messaging

The DeviceNet Communication Protocol is based on the concept of connections method. Master should create connections with slave devices based on the command of exchanging information and I/O data. To establish the master control mechanism, there are only four main steps to be followed. Figure 1.5 demonstrates the basic process for the DeviceNet master communication. The every step function is described in below:

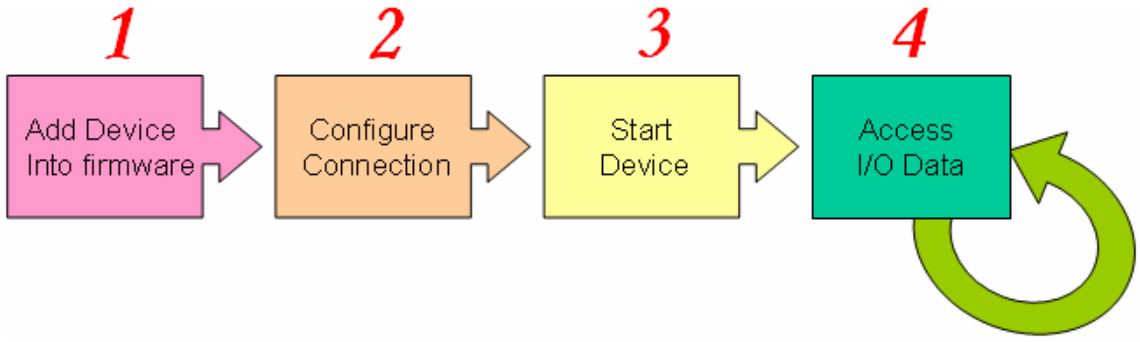


Figure 1.5 Four steps to establish connection

1. **Add device into firmware**

   You should provide the slave device's MAC ID to add into firmware by using API function.

2. **Configure connection**

   You can check the slave device's I/O connection type and the I/O data length. When configuring the I/O connection, you should provide these parameters.

3. **Start Device**

   After configuring connections, users should start device by using API function. The master will communicate with the slave device.

4. **Access I/O data**

   After communicating with slave devices, you can access the I/O data with corresponding read/write function.

After adding the device into the firmware, the master will wait for the I/O configuration information. Then users can create the I/O connections in the next step. Once I/O connections have been created and started, I/O data may be exchanged among devices in the DeviceNet network according to master device demand. Therefore, the master device can access I/O data of the slave devices by one of the four I/O connection methods. The API functions are not only easy to use but also providing a lot of the DeviceNet Master functions to retrieve and deliver the slave's I/O data. For more information, please refer to functions description and demo programs in section 4.

1.5 PISO-DNM100 Firmware Characteristics

The PISO-DNM100 is a high-performance DeviceNet master board. The firmware inside the board implements DeviceNet protocol automatically when the board is active. The firmware always listens to the bus and receives the message at the same time. It works as shown in Figure 1.6.

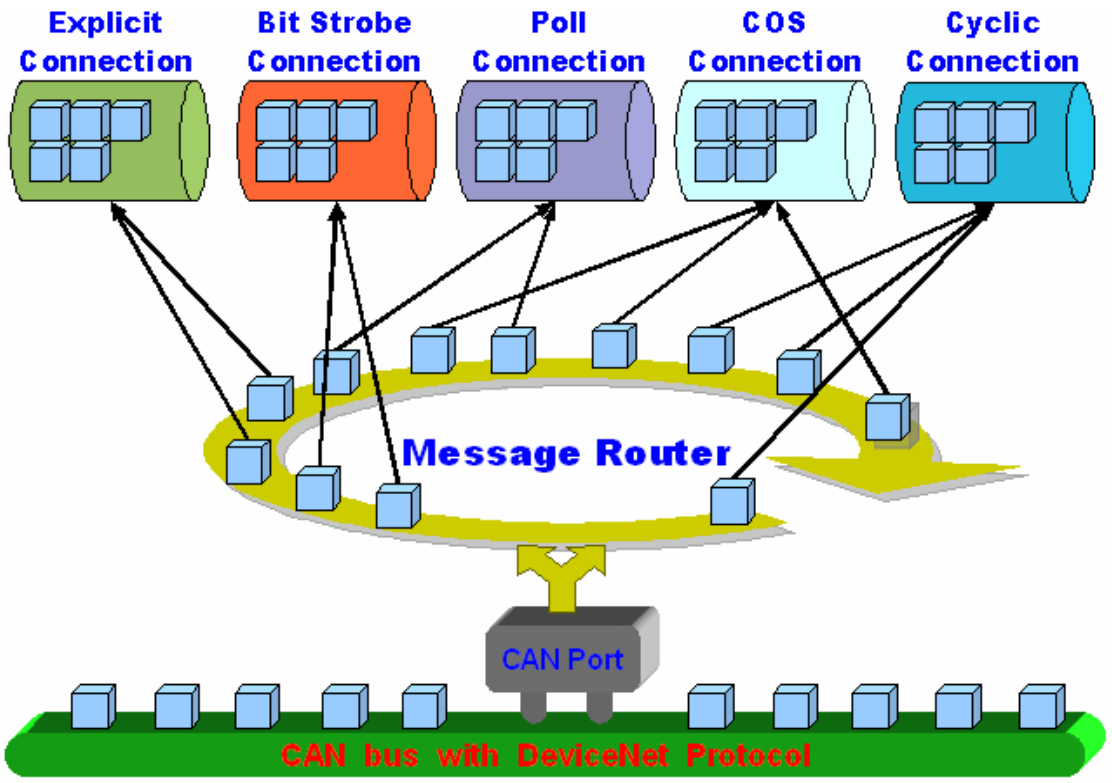


Figure 1.6 Message Router

The PISO-DNM100 firmware has a “ScanList” to store the remote slave devices information. After power off, the information still exists in the EEPROM. When the users turn on the PC next time, the “ScanList” will be loaded from EEPROM. The users can easily use the DLL functions to configure it, including adding devices or removing devices. It works as shown in Figure 1.7. There is more information about the library functions in chapter 4.

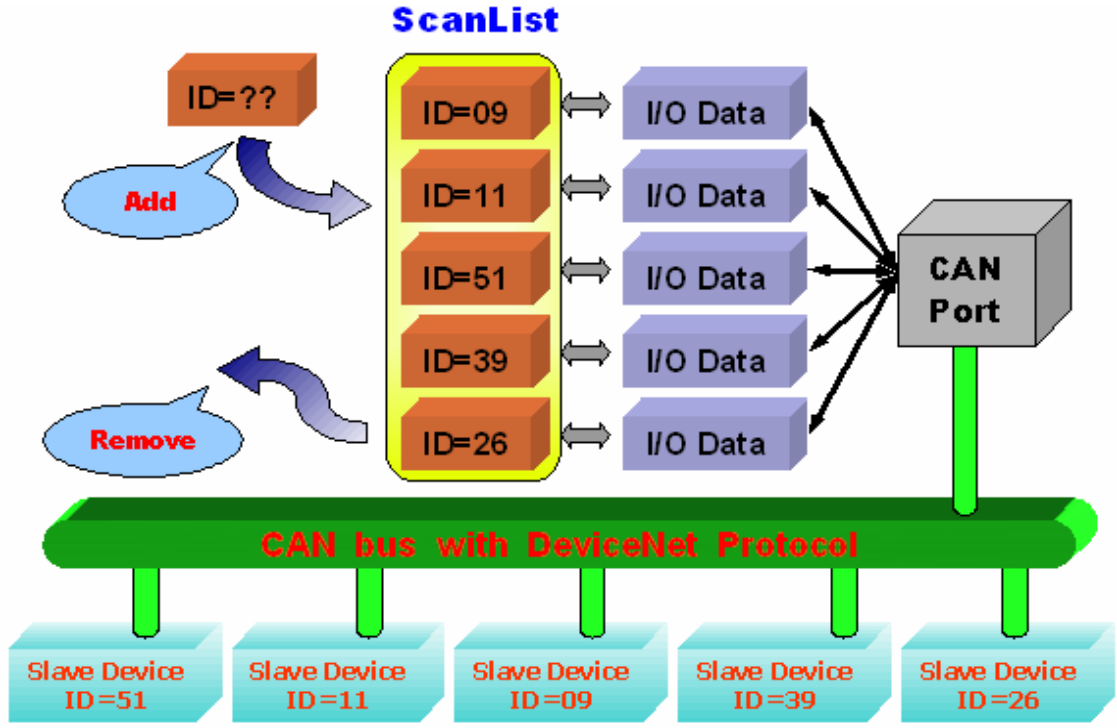


Figure 1.7 ScanList data structure

1.6 Features

Hardware Features

- PCI BUS interface.
- Driver supported for Windows 98/ME/NT/2000/XP.
- 186 compatible CPU with DeviceNet firmware inside.
- 8K bytes DPRAM inside.
- One CAN communication port.
- Compatible with CAN specification 2.0 parts A and B.
- Jumper select 120Ω terminator resistor for each port.
- 2 indicating LED (one for green and another for red).
- Direct memory mapping to the CAN controllers.
- 33MHz 32bit 5V PCI bus (V2.1) plug and play technology.
- 2500Vrms photo-isolation protection on CAN bus.

DeviceNet Firmware Features

- Programmable Master MAC ID.
- Programmable transfer-rate 125K, 250K, 500K.
- Each port support maximum nodes up to 64
- Support Group 2 Only Server functions
- Support UCMM functions
- Predefined Master-Slave Connection Set
- The maximum Fragment number is (Input/Output) up to 64
- Support I/O Operation Mode: Poll, Bit-Strobe and Change Of State/Cyclic
- Support Auto-Scan slave device function.
- Support on-line adding and removing devices.
- Support boot-up auto communicating with slave devices.
- Support Auto-Reconnect when the connection is broken.

1.7 Specifications

- CAN controller: Phillips SJA1000T.
- CAN transceiver: Phillips 82C250/251.
- Signal support: CAN_H, CAN_L.
- CAN controller frequency :16 MHz
- Connector: 5-pin screw terminal connector or 9-pin D-sub male connector.
- Isolation voltage: 2500Vrms on CAN bus.
- 186 compactable CPU
- 8K bytes DPRAM (1K bytes for system)
- 512 K bytes Flash memory (128K bytes for system)
- 512K bytes SRAM
- RTC (real time clock) inside
- 2K EEPROM (256 bytes for system)
- 31 bytes NVRAM
- Power requirements:

  CAN400: 5V@640mA

  CAN200: 5V@380mA
- Environmental:

  Operating temp: 0~60℃

  Storage temp: -20~80℃

  Humidity: 0~90% non-condensing

  Dimensions: 130mm X 110mm

1.8 Block Diagram

The figure 1.8 shows the block diagram of the PISO-DNM100 board.

1. DPRAM (Dual Port RAM) :
   The DPRAM is the memory buffer which provides the communication channel between PC and PISO-DNM100.

2. EEPROM :
   The EEPROM stores the configuration information. After restarting the PC, the configuration data will be loaded form the EEPROM automatically.

3. Control CPU :
   The CPU inside implementing the DeviceNet firmware.

4. CAN Controller :
   The CAN controller is used for sending and receiving the CAN messages. There is photo isolation between CAN controller and CAN bus.

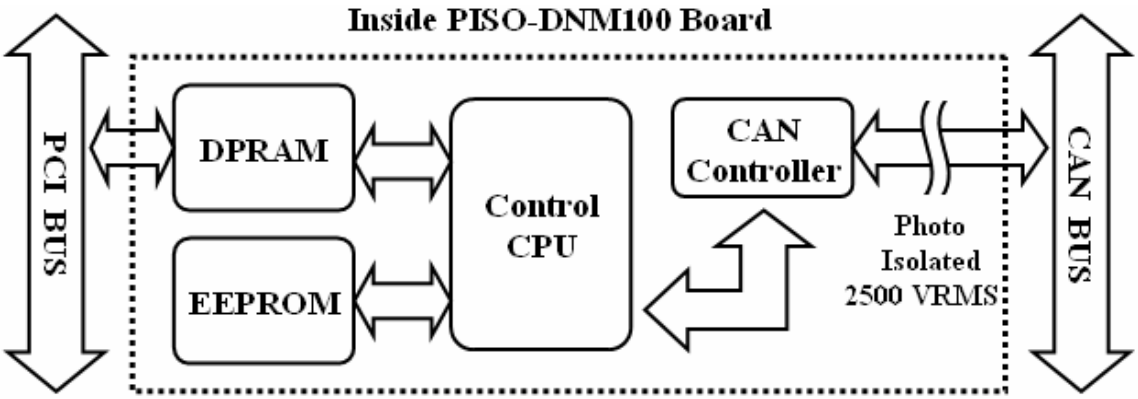


Figure 1.8 Block diagram of the PISO-DNM100

1.9 Product Check List

In addition to this manual, the package includes the following items:

- PISO-DNM100 card;
- Software CD ROM;
- User manual;
- Quick Start manual;
- Release Note

It is recommended that users should read the release note first. All of the important information needed will be provided in the release note as follows:

- Where you can find the software driver, utility and demo programs.
- How to install software & utility.
- Where is the diagnostic program?
- FAQ's and answers.

Attention !

If any of these items are missing or damaged, please contact your local field agent. Keep aside the shipping materials and carton in case you want to ship or store the product in the future.

2. Hardware Configuration

This section will describe the hardware settings of the PISO-DNM100. This information includes the wire connection and terminal resistance configuration for the CAN network.

2.1 Board Layout

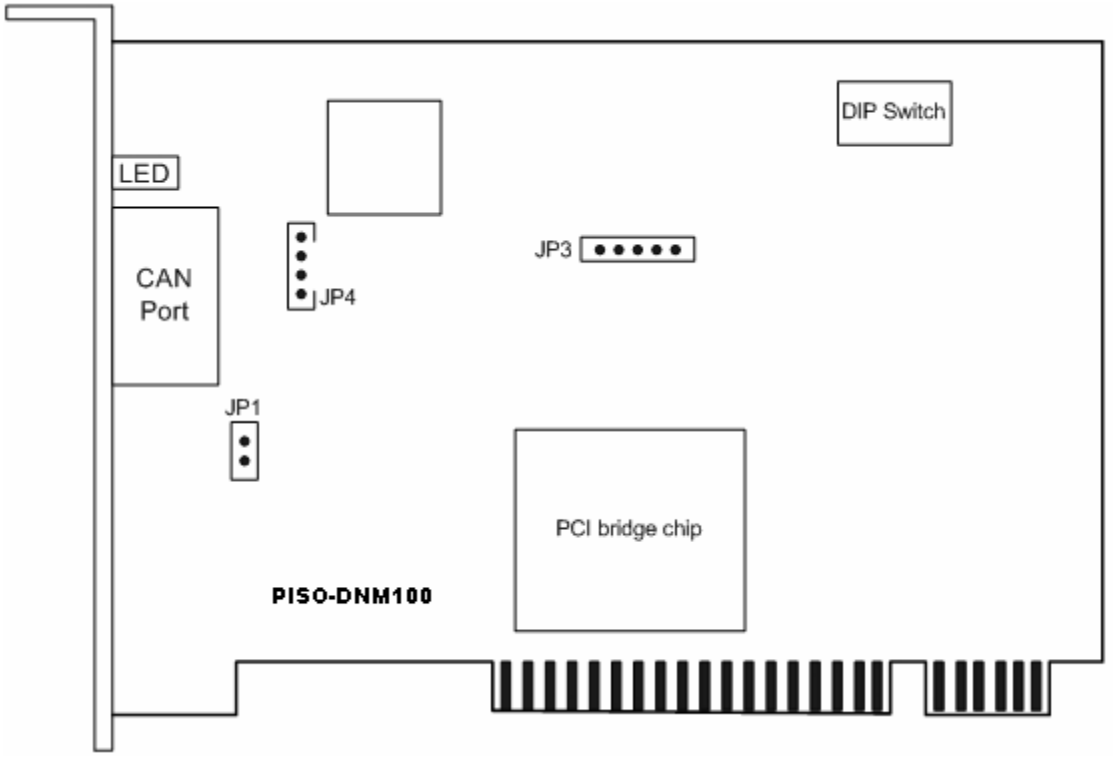


Figure2.1 PISO-DNM100 Board LAYOUT

Note: PISO-DNM100-T layout is similar with PISO-DNM100-D. The only difference is the position of CAN port connector. The positions of jumper or DIP switch are the same. Therefore, users can also refer to the PISO-DNM100-D layout to configure the jumper or DIP switch if they use PISO-DNM100-T.

2.2 Jumper Selection

The following table shows the definition of jumpers or DIP switch. Users need to refer to this table to configure the PISO-DNM100- D/T hardware.

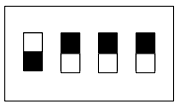
Jumper	Description	Status
JP1	CAN Port 120Ω terminal resistance.	Enable Disable
JP3	Reset pin for download error. If users want to update firmware but the process is fail, users can enable this jumper to reset the PISO-DNM100-D/T into download mode.	Enable Disable
JP4	None.	None
DIP switch	DIP switch is used to set the PISO-CM100 board No. Switch1 is for bit0, switch2 is for bit1 and so forth. For example, if the left-hand-side switch (switch 1) is ON, the board No. is set to 1. The range of board No. is from 0 to 15. Be careful that the board No. for each PISO-CM100-D/T, PISO-DNM100-D/T and PISO-CPM100-D/T must be unique.	 This situation indicates the board No. 1.

Table 2.1 Jumper or DIP switch selections

2.3 Connector Pin Assignment

The PISO-DNM100-T is equipped with one **5-pin screw terminal connector** and the PISO-DNM100-D is equipped with one **9-pin D-sub male connector** for wire connection of the CAN bus. The connector's pin assignment is specified as follows:

2.3.1 5-pin screw terminal connector

The 5-pin screw terminal connector for the CAN bus is shown in Figure 2.4 and the details for the pin assignment are presented in Table 2.2.

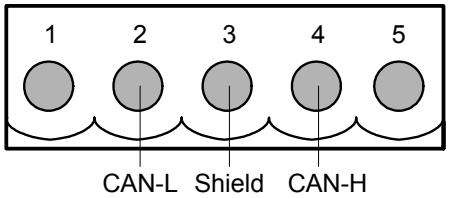


Figure2.4 5-pin screw terminal connector

Pin No.	Signal	Description
1	N/A	No use
2	CAN_H	CAN_H bus line (dominant high)
3	CAN_SHLD	Optional CAN Shield
4	CAN_L	CAN_L bus line (dominant low)
5	N/A	No use

Table 2.2: Pin assignment of 5-pin screw terminal connector

2.3.2 9-pin D-sub male connector

The 9-pin D-sub male connector of the CAN bus interface is shown in Figure 2.5 and the corresponding pin assignments are given in Table 2.3.

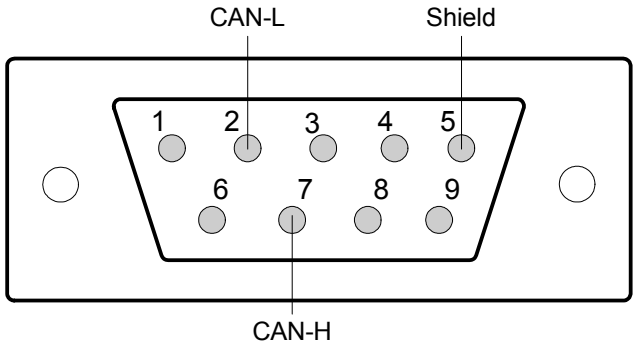


Figure2.5 9-pin D-sub male connector

Pin No.	Signal	Description
1	N/A	No use
2	CAN_L	CAN_L bus line (dominant low)
3	N/A	No use
4	N/A	No use
5	CAN_SHLD	Optional CAN Shield
6	N/A	No use
7	CAN_H	CAN_H bus line (dominant high)
8	N/A	No use
9	N/A	No use

Table 2.3 Pin assignment of the 9-pin D-sub male connector

2.3.3 Wire connection

In order to minimize the reflection effects on the CAN bus line, the CAN bus line has to be terminated at both ends by two terminal resistances as in the following figure. According to the ISO 11898-2 spec, each terminal resistance is 120Ω (or between 108Ω~132Ω). The length related resistance should have 70 mΩ/m. Users should check the resistances of the CAN bus, before they install a new CAN network.

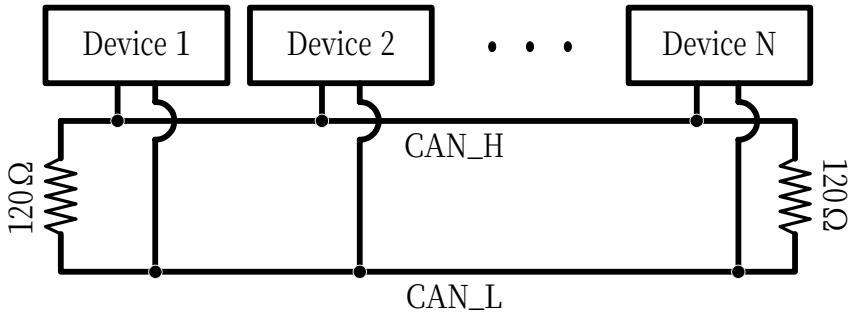


Figure 2.4 CAN bus network topology

Moreover, to minimize the voltage drop over long distances, the terminal resistance should be higher than the value defined in the ISO 11898-2. The following table can be used as a good reference.

Bus Length (meter)	Bus Cable Parameters		Terminal Resistance (Ω)
	Length Related Resistance (mΩ/m)	Cross Section (Type)	
0~40	70	0.25(23AWG)~ 0.34mm^2(22AWG)	124 (0.1%)
40~300	< 60	0.34(22AWG)~ 0.6mm^2(20AWG)	127 (0.1%)
300~600	< 40	0.5~0.6mm^2 (20AWG)	150~300
600~1K	< 20	0.75~0.8mm^2 (18AWG)	150~300

Table 2.4 Relationship between cable characteristics and terminal resistance

2.4 Indicator LED

2.4.1 Green LED

The [Green] LED indicates the firmware status in the PISO-DNM100. There are 3 situations in [Green] LED.

(1). LED off:

This indicates that there are some errors on the bus or in the firmware. The DeviceNet firmware is not running.

(2). LED twinkle:

This indicates that the CAN bus works fine. But there is no any slave devices configuration in the EEPROM of the PISO-DNM100. The DeviceNet firmware is waiting for configuration.

(3). LED on:

This indicates that the DeviceNet firmware is running. The PISO-DNM100 is communicating with the slave devices.

2.4.2 Red LED

The [Red] LED means Network Status. It indicates that there are errors on the bus or there is any slave device's MAC ID collides with the PISO-DNM100's MAC ID. There are two situations in [Red] LED.

(1). LED off:

This indicates that there is no error on the bus and about the MAC ID.

(2). LED twinkle:

This indicates that there are errors on the bus which maybe the situations as shown bellow:

(a) The CAN connector doesn't connect to the slave devices.
(b) The power of the slave devices is off.
(c) The MAC ID collision between master and slave devices is occurring.

2.5 Hardware Installation

When users want to use PISO-DNM100-D/T, the hardware installation needs to be finished as following steps.

1. Shutdown your personal computer.
2. Configure the DIP switch and JP1 of your PISO-DNM100-D/T for board No. and terminal resistance. More detail information could be found on the figure 2.1 and table 2.1.
3. Check JP3 status of PISO-DNM100-D/T. If necessary, enable it.
4. Find an empty PCI slot for your PISO-DNM100-D/T on the mother board of the personal computer. Plug the configured PISO-DNM100-D/T into this empty PCI slot. See figure 2.5.
5. Plug your CAN bus cable(s) into the 5-pin screw terminal connector or the 9-pin D-sub connector.

When the steps described above is completed, turn on the personal computer.

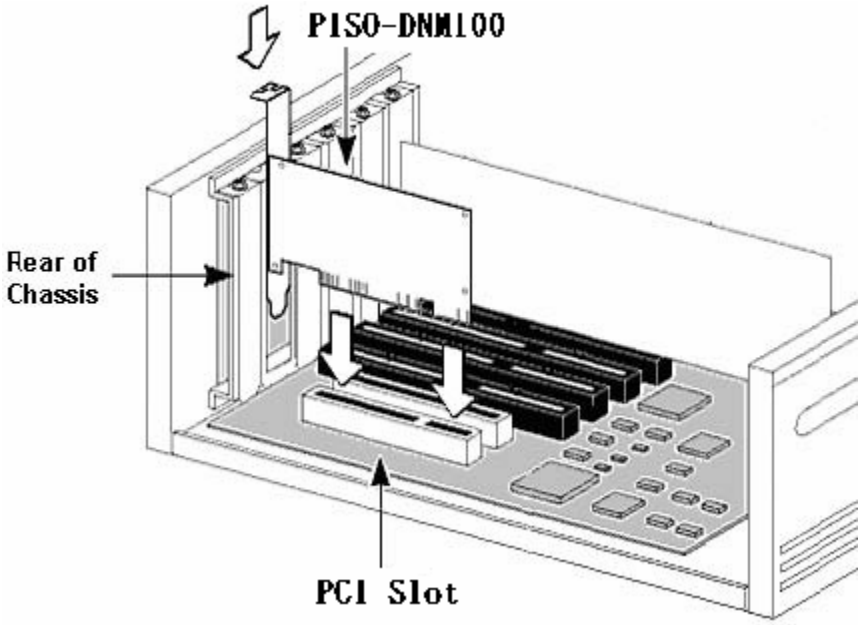


Figure 2.5 PISO-DNM100 installation

3. Driver Installation and Software Application

The DeviceNet DLL driver (DNM100.dll) collection of function calls for the PISO-DNM100 cards used in Windows 98/Me/NT/2000/XP systems. The application structure is presented in the following figure. The user's DeviceNet application programs can be developed by the following designated tools: VB, Delphi and Borland C++ Builder…etc. In these tools, the application program can call the DNM100.DLL driver to implement DeviceNet network application. And then the DeviceNet DLL driver will throughout the CM100.dll into the KP_CM100.sys and windrvr6.sys to access the hardware system, as shown in the following Figure.

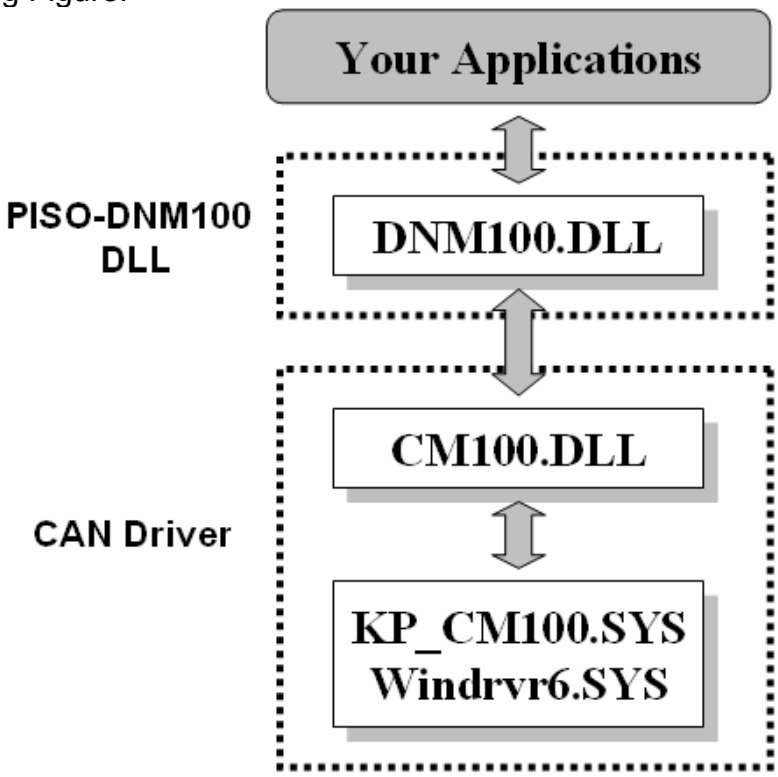


Figure 3.1 Software architecture in the Windows system

In the following sub-section, we show some flow diagrams to describe how to apply the DeviceNet protocol (DNM100.DLL) to build a master device. Section 3.2 ~ 3.10 show the flow diagram for users to understand easily. Note that users need to follow the operation principle of the DeviceNet protocol correctly and easily to communicate with the remote nodes by these connection methods.

3.1 Driver Installation of the PISO-DNM100

The software Installation for DeviceNet application is demonstrated as the following descriptions. After finishing the procedure, the driver, demos, manual and Utility can be in your PC. For the advance application, users can refer to the basic demo programs to develop the customized DeviceNet master application.

The driver of PISO-DNM100 can be used in 98/Me/NT/2000/XP Windows environments. For these Windows operation systems, the recommended installation procedure is given as follows:

Step 1: Insert the companion CD into the CD-ROM driver and wait a few seconds until the installation program starts automatically. If it cannot be started automatically for some reason, please double-click the file **ICPDAS.EXE** on this CD. The screenshot likes the following picture. Please click “CAN Series” item.

Step 2: After changing to the following picture, please click the “DeviceNet” item.

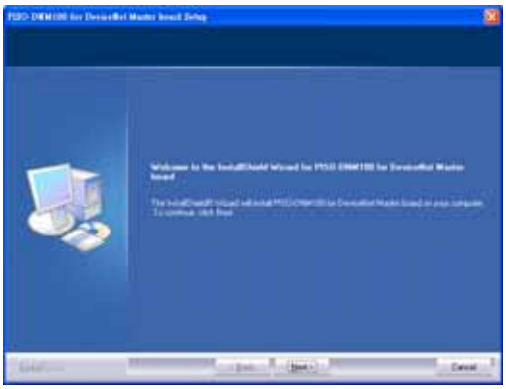
Step 3: When the window is changing to the picture below, please move the mouse cursor on the “Master” item.

Step 4: The DeviceNet master products will be shown in the window.

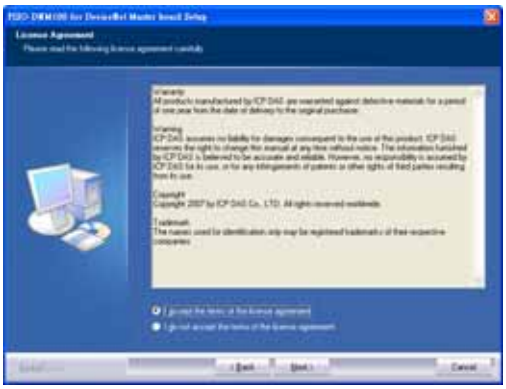
Step 5: Click “install Toolkit for Windows 98, Me, NT, 2000, XP, which is based on the operation system you used”.

Step 6: After finishing the step 5, the users will see the setup file in the folder “PISO-DNM100_V10.exe”. Please double-click it to run the setup.

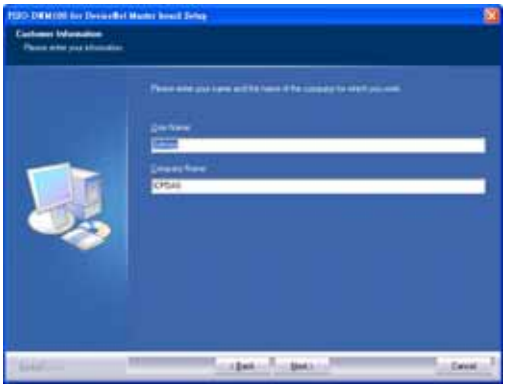
Step 7: This first setup screen shoot is shown as follows. Please press “Next” button to continue the process.



Step 8: Press “Next” button. The screen shoot is shown below. After reading the license, the users can accept it or not. If the users accept it, please select “I accept….” and press “Next” button.

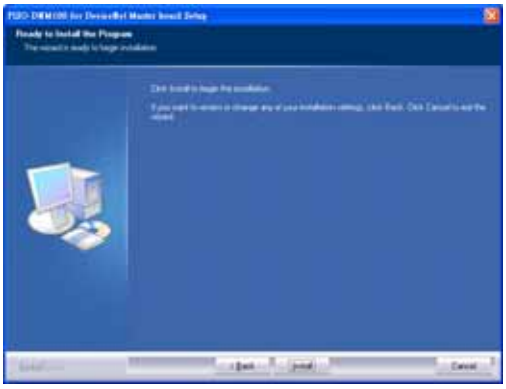


Step 9: After accepting the license, the next screen shoot is shown as follows. Users can edit your name and company name. After editing the information, please press “Next” button.

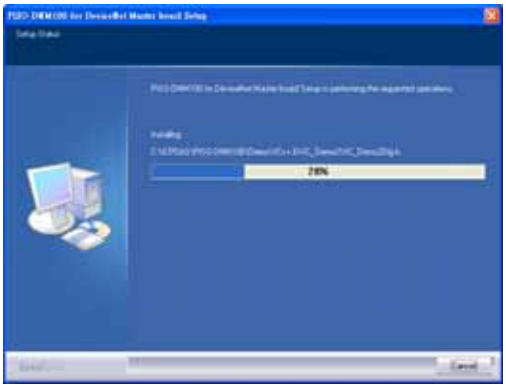


Step 10: After editing the information, the next screen shoot is shown as follows. Please select “Complete” item and press “Next” button.

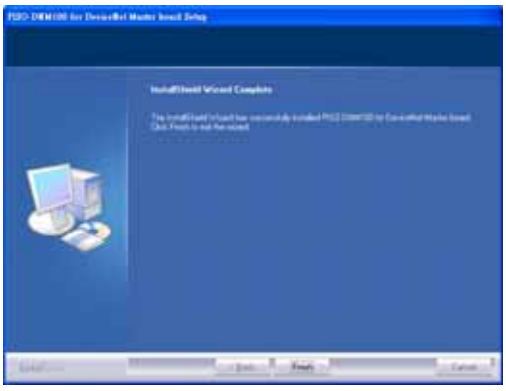
Step 11: The next screen shoot is shown as follows. Please press “Install” button. The setup process will start.



Step 12: The setup process is running. The screen shoot is shown below.



Step 13: Wait for the setup process finishing. The next screen shoot is shown below. After finishing the process, please press “Finish” button.



Step 14: The next screen shoot is shown as follows. Please restart your PC.

Then the setup software would copy the related material to the indicated directory and register the driver on your computer. The driver target directory is different according to the different systems as follows.

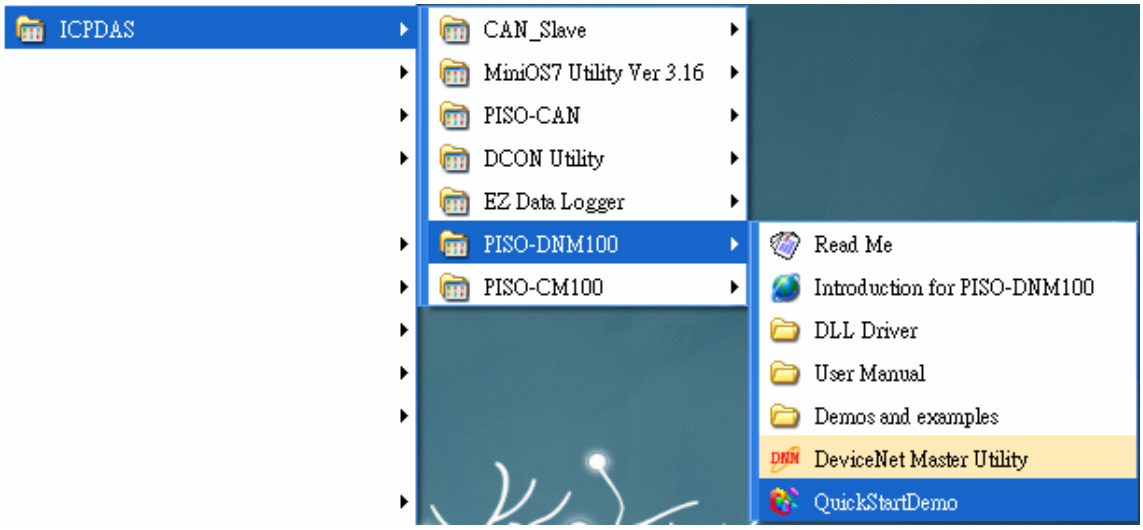
Windows NT/2000 – **WINNT\SYSTEM32\DRIVERS**

Windows 98/Me/XP – **WINDOWS\SYSTEM32\DRIVERS**

The other data and resource is copied to the following directory:

C:\ICPDAS\PISO-DNM100

The program files picture is shown as follow.



Note：DeviceNet Master Utility is a useful tool for users to configure and test the DeviceNet slave devices. You can refer to DeviceNet Master Utility manual to know the detail. The next sections in this manual explain how to implement the DeviceNet system by API functions.

3.2 Flow Diagram for Debugging

Before developing the DeviceNet applications, users should test to connect to the slave devices. When users have no idea to communicate with them, users can follow these steps shown in figure 3.2. The following functions can help users to get the DeviceNet information of the slave devices. The users can choose one of them to find out the problem of the slave devices. The detail information about those functions is in the next chapter.

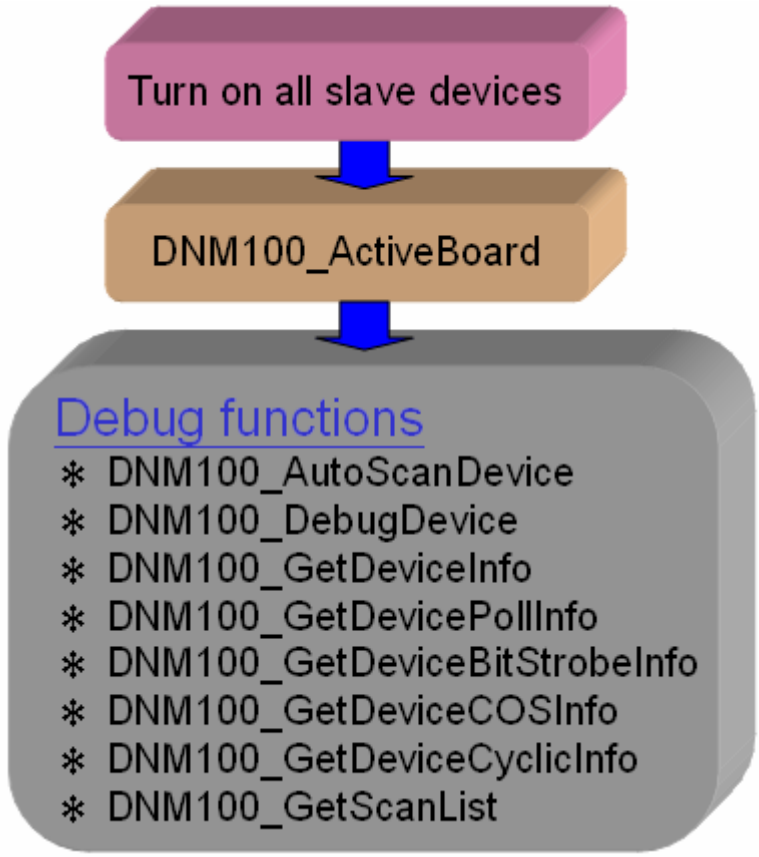


Figure 3.2 Debugging Diagram

3.3 Flow Diagram for I/O Configuration

After getting the DeviceNet I/O information of the slave devices, users should save the parameters into the EEPROM in PISO-DNM100. The EEPROM will store the configuration data. The firmware in PISO-DNM100 will load the previous configuration from the EEPROM in the next boot-up. When the devices in the DeviceNet network are changed, the users must set the configuration data to fit the application. The configuration diagram is shown in Figure 3.3. There is more information about those functions in the next chapter.

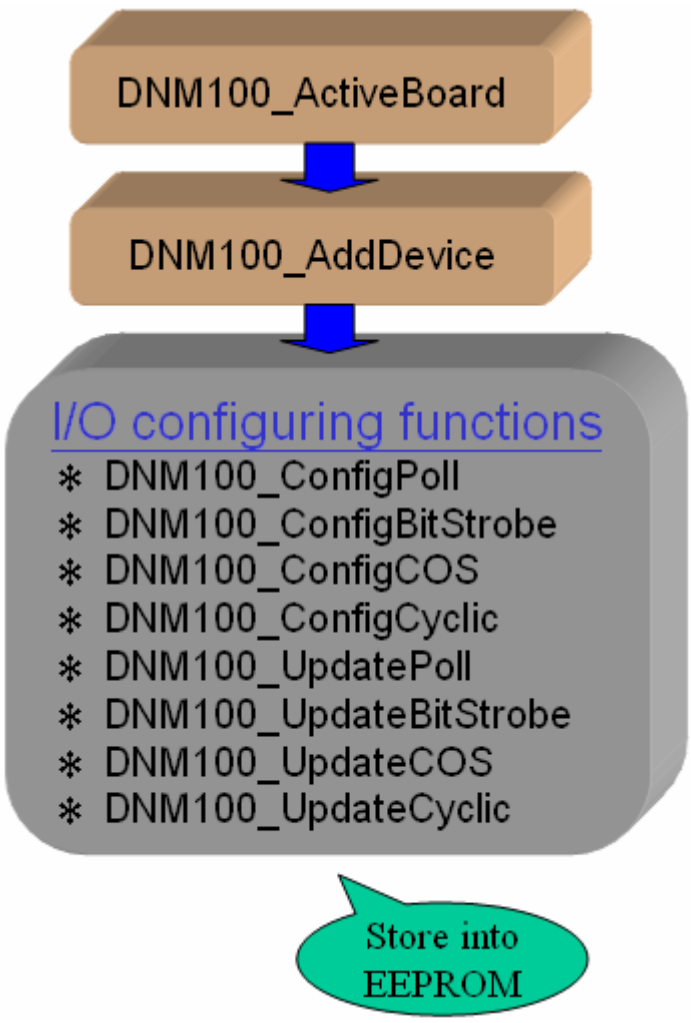


Figure 3.3 I/O configuration Diagram

3.4 Flow Diagram for General I/O Operation

After configuring the PISO-DNM100, the users can easily read or write I/O data from or to the remote DeviceNet slave devices. The users don't need to know about the DeviceNet protocol. The main steps are shown in Figure 3.5. There are more detail descriptions in chapter 3.6 ~ 3.10.

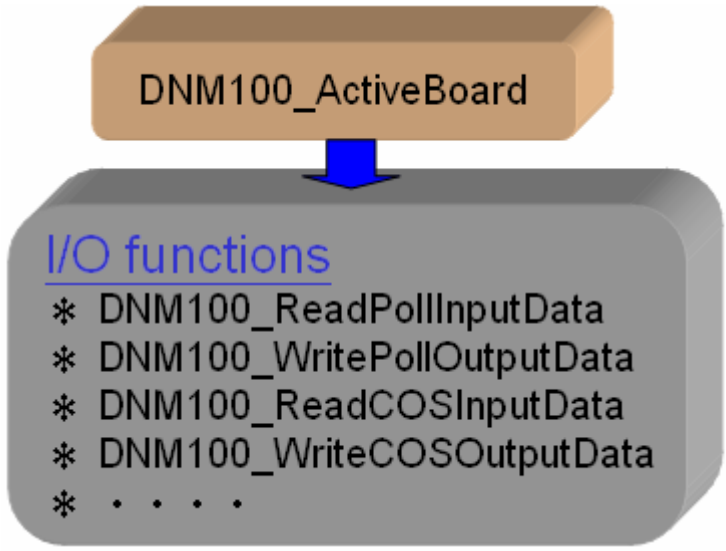


Figure 3.5 General I/O Diagram

3.5 Flow Diagram for On-line Adding/Removing Device

The PISO-DNM100 provides the on-line adding/removing slave device functions. The users can follow the steps to achieve this function. The steps are shown in Figure 3.6 and Figure 3.7.

1. On-line Adding Devices :

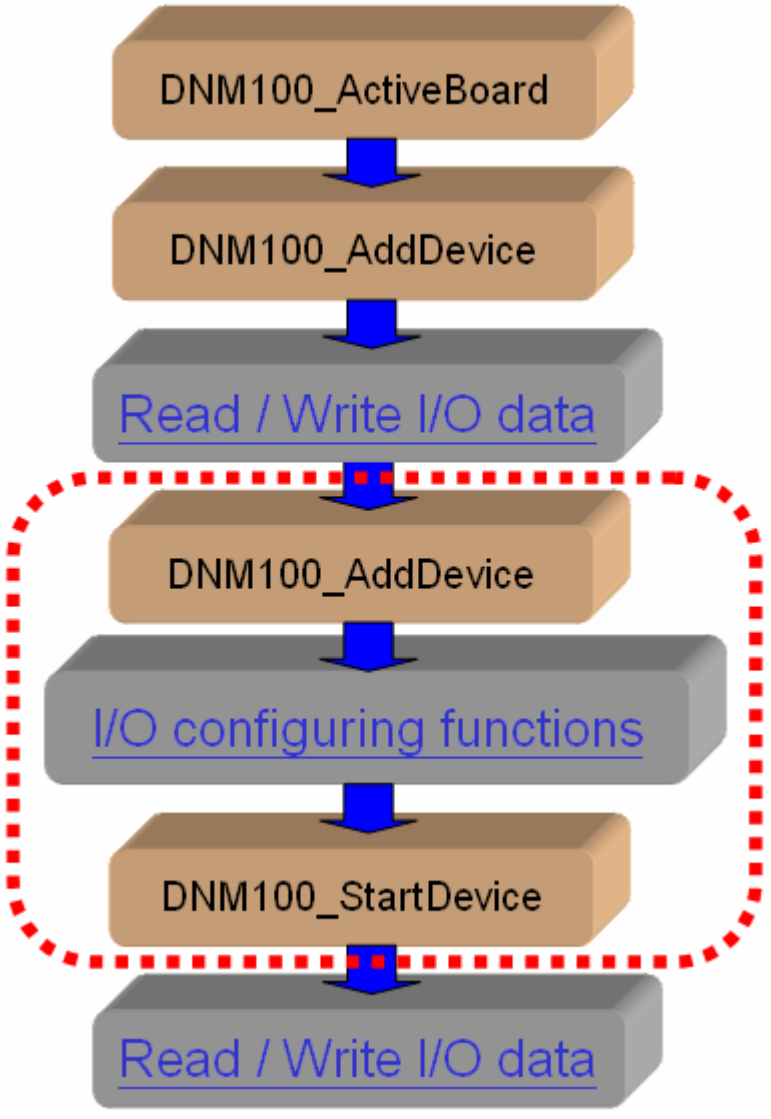


Figure 3.6 On-line Add Device Diagram

2. On-line Removing Devices :

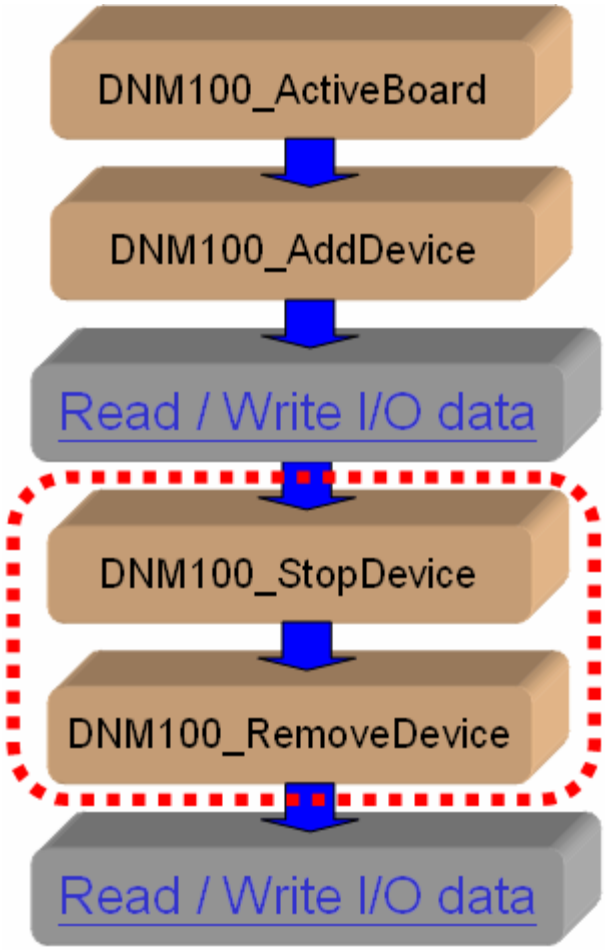


Figure 3.7 On-line Remove Device Diagram

3.6 Flow Diagram for “SetAttribute” and “GetAttribute”

The users can set or get DeviceNet device's property via DeviceNet network. The PISO-DNM100 provides these functions to set or get the properties of the remote devices easily. The steps are shown in Figure 3.8.

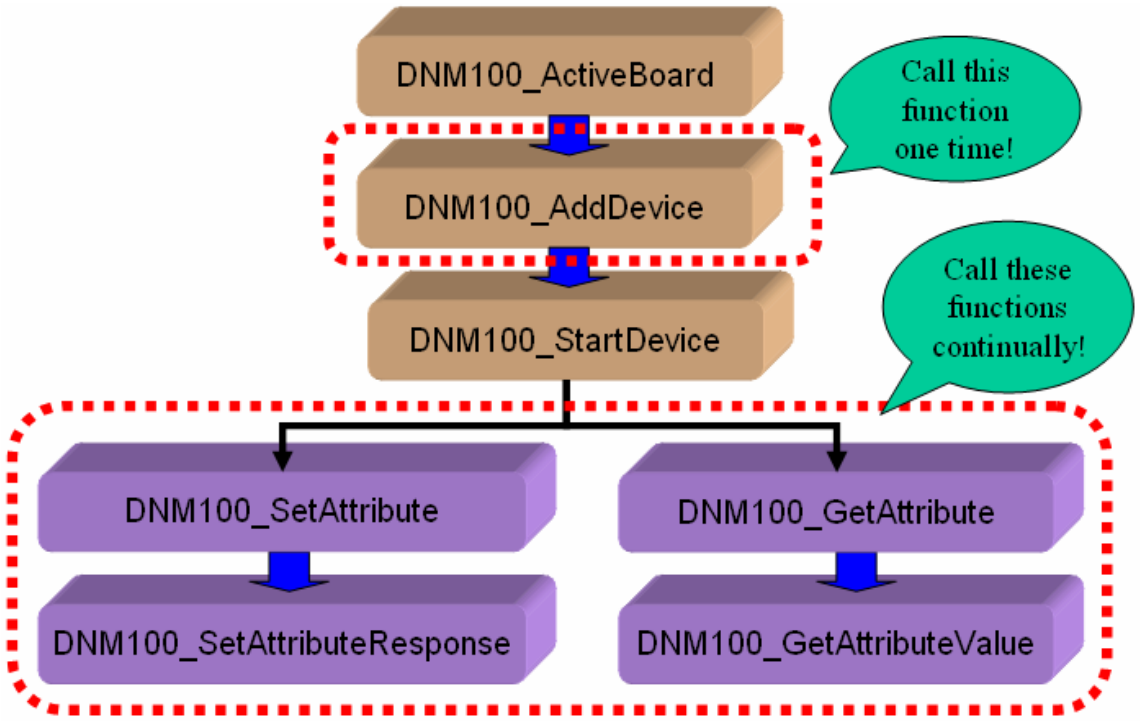


Figure 3.8 “SetAttribute” and “GetAttribute” Diagram

3.7 Flow Diagram for Poll Connection

The users can read or write device's I/O data via the DeviceNet Poll connection. The PISO-DNM100 provides three functions to read and write the I/O data easily. Firstly, the users should know the device's I/O input length (in Byte) and output length (in Byte). Secondly, the users should set these two parameters by calling DNM100_ConfigPoll. Thirdly, the users can set the initial output value by calling DNM100_WritePollOutputData before starting the specific slave device. If the users do not initialize the output value, the firmware default output value is 0. Fourthly, the users can start communicating with device to read or write I/O data. If the specific slave device doesn't have any output channel, the firmware will start communicating with the device automatically. The Figure 3.9 shows the main steps to achieve this function. There are more functions described in chapter 4.

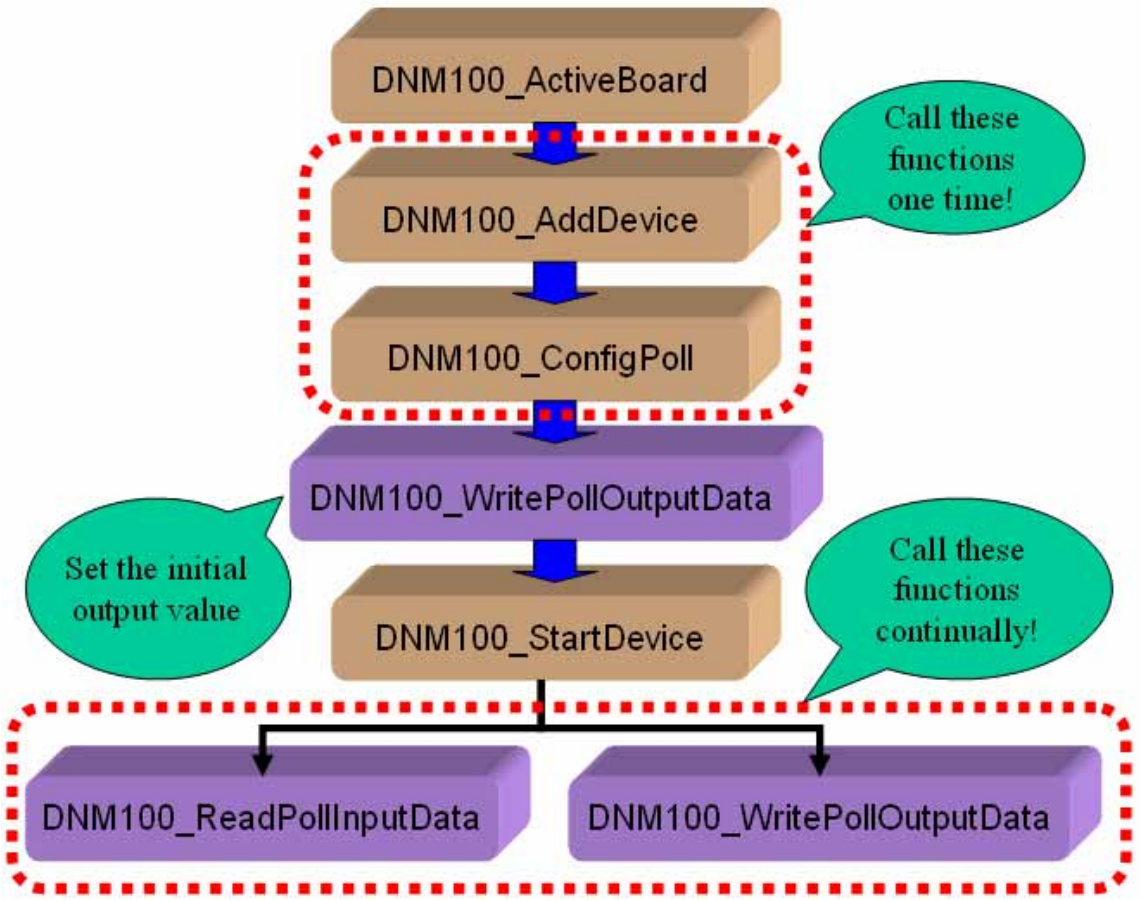


Figure 3.9 Poll Connection Diagram

3.8 Flow Diagram for Bit-Strobe Connection

The users can read device's I/O data via DeviceNet Bit-Strobe connection. The PISO-DNM100 provides two functions to read the I/O data easily. Firstly, the users should know the device's I/O input length (in Byte). Secondly, the users should set this parameter by calling DNM100_ConfigBitStrobe. Thirdly, the users can start communicating with device to read I/O data. The Figure 3.10 shows the main steps to achieve this function. There are more functions described in chapter 4.

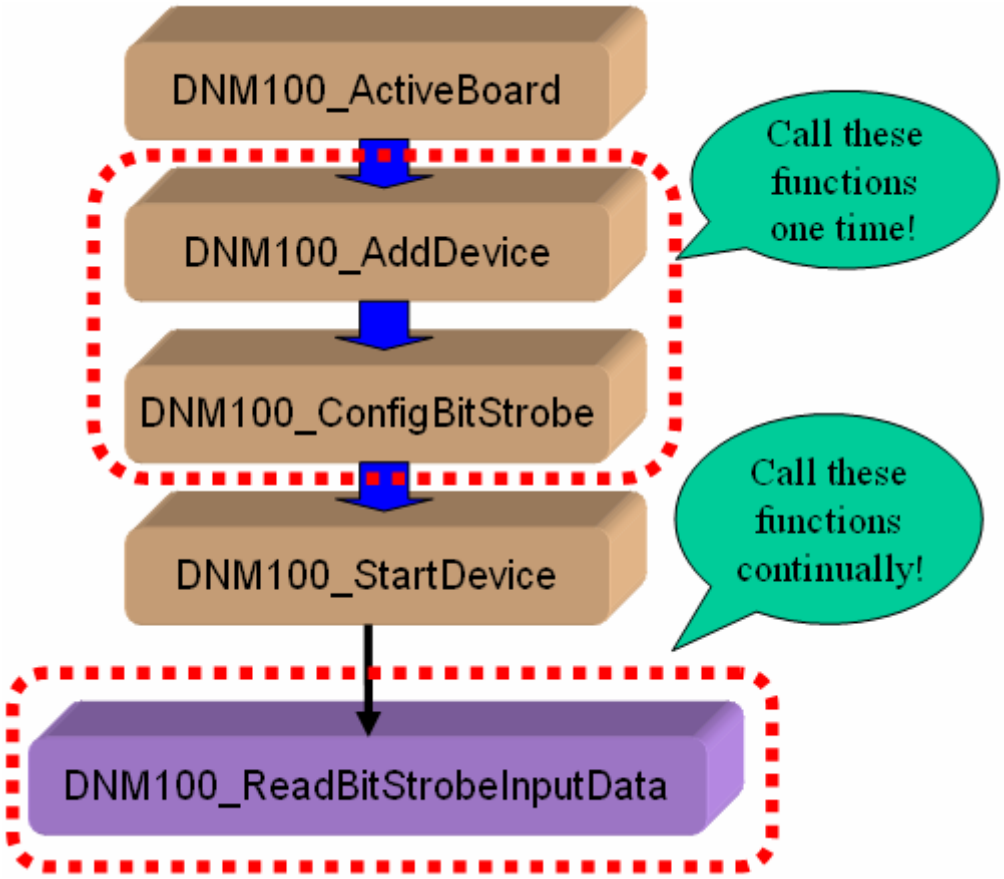


Figure 3.10 Bit-Strobe Connection Diagram

3.9 Flow Diagram for COS Connection

The users can read or write device's I/O data via DeviceNet COS (change-Of-State) connection. The PISO-DNM100 provides three functions to read or write the I/O data easily. Firstly, the users should know the device's I/O input length (in Byte) and the output length (in Byte). Secondly, the users should set this parameter by calling DNM100_ConfigCOS. Thirdly, the users can set the initial output value by calling DNM100_WriteCOSOutputData before starting the specific slave device. If the users do not initialize the output value, the firmware default output value is 0. Fourthly, the users can start communicating with device to read or write I/O data. If the specific slave device doesn't have any output channel, the firmware will start communicating with the device automatically. The Figure 3.11 shows the main steps to achieve this function. There are more functions described in chapter 4.

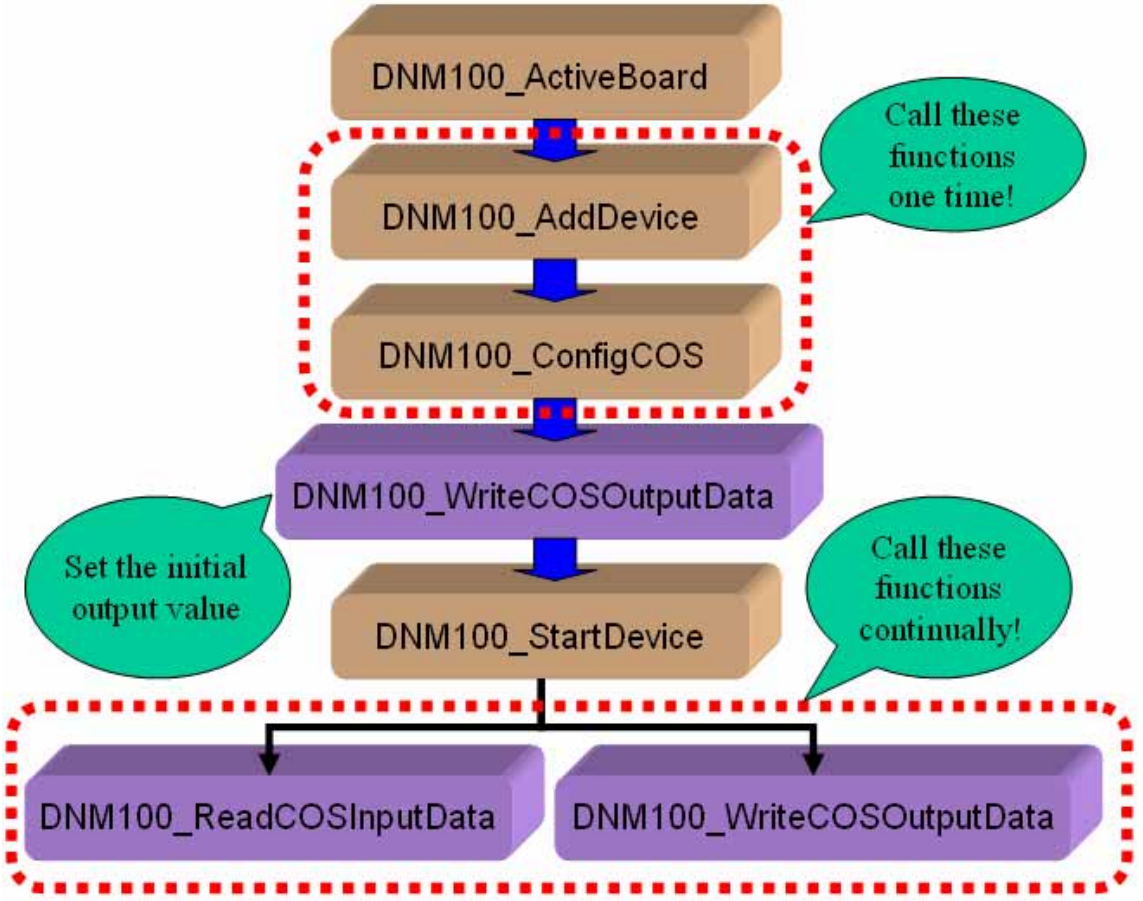


Figure 3.11 COS Connection Diagram

3.10 Flow Diagram for Cyclic Connection

The users can read or write device's I/O data via DeviceNet Cyclic connection. The PISO-DNM100 provides three functions to read or write the I/O data easily. Firstly, the users should know the device's I/O input length (in Byte) and the output length (in Byte). Secondly, the users should set this parameter by calling DNM100_ConfigCyclic. Thirdly, the users can set the initial output value by calling DNM100_WriteCyclicOutputData before starting the specific slave device. If the users do not initialize the output value, the firmware default output value is 0. Fourthly, the users can start communicating with device to read or write I/O data. If the specific slave device doesn't have any output channel, the firmware will start communicating with the device automatically. The Figure 3.12 shows the main steps to achieve this function. There are more functions described in chapter 4.

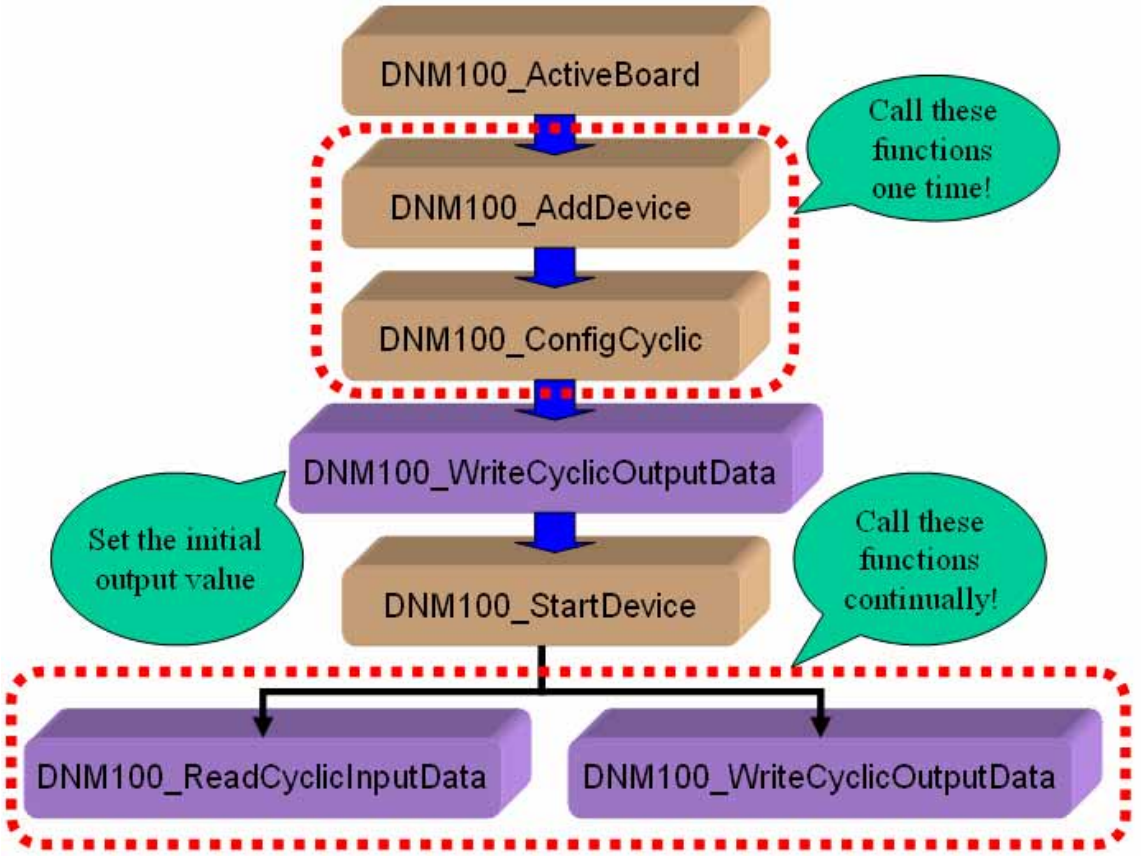


Figure 3.12 Cyclic Connection Diagram

4. Function description

All the functions of the PISO-DNM100 can be separated into five groups. The idea is shown Figure 4.1. There is more detail description in CH 4.1.

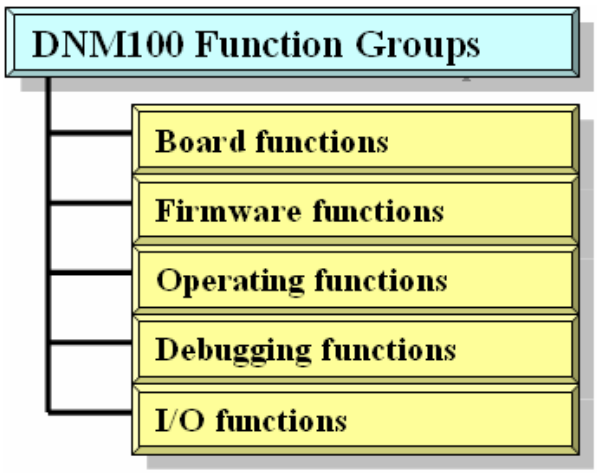


Figure 4.1 Five Function Groups

[Board Functions]

These functions in this group help users to find DNM100 boards or get board's information. The users can use these functions to configure or manage the boards in the PC.

[Firmware Functions]

These functions in this group help users to operate the firmware or get the status of the firmware inside the PISO-DNM100.

[Operating Functions]

These operating functions are the important operation of the DeviceNet master. They help users to configure the whole network.

[Debugging Function]

These debugging functions can help user to debug the network, including the wire connection, the slave device's setting…etc. When building the DeviceNet network, the user can use these functions to make sure that the network or the slave devices are fine.

[I/O Functions]

These functions help user to read or to write the I/O data from or to the remote slave devices.

4.1 DLL Function Definition and Description

All the functions provided in the DNM100.dll are listed in the following table and detail information for every function is presented in the next sub-section. However, in order to make the descriptions more simply and clear, the attributes for the both the input and output parameter functions are given as **[input]** and **[output]** respectively, as shown in following table.

Keyword	Set parameter by user before calling this function?	Get the data from this parameter after calling this function?
[input]	Yes	No
[output]	No	Yes

Table 4.1.1 Functions Table (Board Functions) 1/1

No.	Function Name	Description
1	DNM100_GetBoardInf	Get the information of the PISO-DNM100
2	DNM100_TotalDNM100Board	Get total PISO-DNM100 boards in the PC
3	DNM100_ActiveBoard	Make PISO-DNM100 active
4	DNM100_CloseBoard	Close the PISO-DNM100
5	DNM100_GetDLLVersion	Get the DLL version of the DNM100.DLL

Table 4.1.2 Functions Table (Firmware Functions) 1/1

No.	Function Name	Description
1	DNM100_GetFirmwareVersion	Get the version of the firmware inside the PISO-DNM100
2	DNM100_ResetFirmware	Reset the firmware in the PISO-DNM100
3	DNM100_CheckFirmwareMode	Get the firmware mode at present

Table 4.1.3 Functions Table (Operating Functions) 1/2

No.	Function Name	Description
1	DNM100_SetMasterMACID	Set the MAC ID of the PISO-DNM100 (DeviceNet Master's MAC ID)
2	DNM100_GetMasterMACID	Get the MAC ID of the PISO-DNM100 (DeviceNet Master's MAC ID)
3	DNM100_GetBaudRate	Get the baud rate of the CAN bus
4	DNM100_SetBaudRate	Set the baud rate of the CAN bus
5	DNM100_GetMasterStatus	Get the status of the PISO-DNM100 (DeviceNet Master's status) at present
6	DNM100_StartDevice	PISO-DNM100 will start to communicate with the specific slave device
7	DNM100_StopDevice	PISO-DNM100 will stop to communicate with the specific slave device
8	DNM100_StartAllDevice	PISO-DNM100 will start to communicate with all slave devices
9	DNM100_StopAllDevice	PISO-DNM100 will stop to communicate with all slave devices
10	DNM100_AddDevice	Add the specific slave device's information into the PISO-DNM100(DeviceNet Master)
11	DNM100_RemoveDevice	Remove the specific slave device's information from the PISO-DNM100 (DeviceNet Master)
12	DNM100_RemoveIOConnection	Remove specific slave device's I/O information from the PISO-DNM100 (DeviceNet Master)
13	DNM100_ConfigPoll	Add Poll I/O information of the specific slave device into the PISO-DNM100 (DeviceNet Master)
14	DNM100_UpdatePollConfig	Update Poll I/O configuration of the specific slave device in the PISO-DNM100 (DeviceNet Master)
15	DNM100_ConfigBitStrobe	Add Bit-Strobe I/O information of the specific slave device into the PISO-DNM100 (DeviceNet Master)

Table 4.1.4 Functions Table (Operating Functions) 2/2

No.	Function Name	Description
16	DNM100_UpdateBitStrobeConfig	Update Bit-Strobe I/O configuration of the specific slave device in the PISO-DNM100 (DeviceNet Master)
17	DNM100_ConfigCOS	Add COS I/O information of the specific slave device into the PISO-DNM100 (DeviceNet Master)
18	DNM100_UpdateCOSConfig	Update COS I/O configuration of the specific slave device in the PISO-DNM100 (DeviceNet Master)
19	DNM100_ConfigCyclic	Add Cyclic I/O information of the specific slave device into the PISO-DNM100 (DeviceNet Master)
20	DNM100_UpdateCyclicConfig	Update Cyclic I/O configuration of the specific slave device in the PISO-DNM100 (DeviceNet Master)
21	DNM100_GetLastDeviceError	Get the slave device's MAC ID which is in error status at present
22	DNM100_GetDeviceStatus	Get the slave device's communication status.
23	DNM100_SetBreakOption	Set the option when the communication is break.
24	DNM100_GetBreakOption	Get the option when the communication is break.
25	DNM100_ClearAllConfig	Clear all configuration in PISO-DNM100

Table 4.1.5 Functions Table (Debugging Functions) 1/1

No.	Function Name	Description
1	DNM100_DebugDevice	PISO-DNM100 will check the specific slave device whether the device is exists or not.
2	DNM100_GetDeviceInfo	Get all the I/O information of the specific slave device
3	DNM100_GetDevicePollInfo	Get the Poll information of the specific slave device
4	DNM100_GetDeviceBitStrobeInfo	Get the Bit-Strobe information of the specific slave device
5	DNM100_GetDeviceCOSInfo	Get the COS information of the specific slave device
6	DNM100_GetDeviceCyclicInfo	Get the Cyclic information of the specific slave device
7	DNM100_AutoScanDevice	PISO-DNM100 will scan the whole network to find out how many devices in the DeviceNet network.
8	DNM100_GetScanList	Get all the I/O information in the firmware

Table 4.1.6 Functions Table (I/O Functions) 1/2

No.	Function Name	Description
1	DNM100_GetExplicitStatus	Get the Explicit connection status of the slave device
2	DNM100_GetExplicitResult	Get the Explicit connection result of the slave device
3	DNM100_GetAttribute	Send the get attribute command to the slave device.
4	DNM100_GetAttributeValue	Get the attribute value of the DNM100_GetAttribute
5	DNM100_SetAttribute	Send the set attribute command to the slave device.
6	DNM100_SetAttributeResponse	Get the response value of the DNM100_SetAttribute
7	DNM100_ReadPollInputData	Read the input data via Poll connection
8	DNM100_WritePollOutputData	Write the output data via Poll connection
9	DNM100_GetPollStatus	Get the Poll connection status of the slave device
10	DNM100_GetPollResult	Get the Poll connection result of the slave device
11	DNM100_CheckPollConnectionStatus	Check the Poll connection status is fine or not
12	DNM100_ReadBitStrobeInputData	Read the input data via Bit-Strobe connection
13	DNM100_GetBitStrobeStatus	Get the Bit-Strobe connection status of the slave device
14	DNM100_GetBitStrobeResult	Get the Bit-Strobe connection result of the slave device
15	DNM100_CheckBitStrobeConnectionStatus	Check the Bit-Strobe connection status is fine or not

Table 4.1.7 Functions Table (I/O Functions) 2/2.

No.	Function Name	Description
16	DNM100_ReadCOSInputData	Read the input data via COSI connection
17	DNM100_WriteCOSOutputData	Write the output data via COSI connection
18	DNM100_GetCOSStatus	Get the COS connection status of the slave device
19	DNM100_GetCOSResult	Get the COS connection result of the slave device
20	DNM100_CheckCOSConnectionStatus	Check the COS connection status is fine or not
21	DNM100_ReadCyclicInputData	Read the input data via Cyclicl connection
22	DNM100_WriteCyclicOutputData	Write the output data via Cyclicl connection
23	DNM100_GetCyclicStatus	Get the Cyclic connection status of the slave device
24	DNM100_GetCyclicResult	Get the Cyclic connection result of the slave device
25	DNM100_CheckCyclicConnectionStatus	Check the Cyclic connection status is fine or not

4.2 Function Return Code

Table 4.2.1 Interpretation of the return code (Hardware Error) 1/1

Return Code	Error ID	Comment
0	DNM100_NoError	No error
10001	DNM100_DriverError	Kernel driver is not opened.
10002	DNM100_ActiveBoardError	This board can not be activated.
10003	DNM100_BoardNumberError	The Board number exceeds the total board numbers.
10004	DNM100_PortNumberError	The Port number is not correct.
10007	DNM100_InitError	The PISO-DNM100 replies error.
10021	DNM100_SoftBufferIsEmpty	No CAN messages in the buffer.
10022	DNM100_SoftBufferIsFull	The software buffer is overflow.
10023	DNM100_TimeOut	The PISO-DNM100 has no response.
10024	DNM100_SetCyclicMsgFailure	The cyclic messages are over 5 counts. This is special function for CAN.
10025	DNM100_DpramOverRange	The command length is over 512 bytes.
10026	DNM100_NoDpramCmd	There is no command in DPRAM.
10027	DNM100_ModeError	This board can't be changed to firmware mode.
10030	DNM100_NoFileInside	There is no firmware in PISO-DNM100.
10031	DNM100_DownloadFailure	The download firmware process is failure.
10032	DNM100_EEPROMDamage	The EEPROM is out of order.
10033	DNM100_NotEnoughSpace	The firmware is too large to put it into the PISO-DNM100-D/T
10034	DNM100_StillDownloading	The firmware is downloading.
10035	DNM100_BoardModeError	The firmware mode is error.
10036	DNM100_CardTypeError	The firmware is not for the PISO-DNM100

Table 4.2.2 Interpretation of the return code (MapTable Error) 1/1

Return Code	MapTable Error	Comment
1200	DNMXS_MapTableFull	The MapTable is full.
1201	DNMXS_MapDataDuplicate	The data already exists in the MapTable.
1202	DNMXS_MapDataNotFound	The data can't be found in the MapTable.
1203	DNMXS_MapTableError	The MapTable has some errors.

Table 4.2.3 Interpretation of the return code (DeviceNet Error) 1/1

Return Code	DeviceNet Error	Comment
1300	DNMXS_BoardNotOnline	The master is not on line.
1301	DNMXS_ExplicitNotEstablish	The Explicit connection is not established.
1302	DNMXS_PollNotEstablish	The Poll connection is not established.
1303	DNMXS_BitStrobeNotEstablish	The Bit-Strobe connection is not established.
1304	DNMXS_COSNotEstablish	The COS connection is not established.
1305	DNMXS_CyclicNotEstablish	The Cyclic connection is not established.

Table 4.2.5 Interpretation of the return code (EEPROM Error) 1/1

Return Code	EEPROM Error	Comment
1400	DNMXS_ReadEEPROMError	There are errors when reading EEPROM.
1401	DNMXS_WriteEEPROMError	There are errors when writing EEPROM.
1402	DNMXS_EEPROMDataError	The data in EEPROM is error.
1403	DNMXS_EEPROMFull	The EEPROM is full.
1404	DNMXS_DeviceNotExistInEEP	The data can't be found in the EEPROM.
1405	DNMXS_EEPROMDeviceDuplicate	The device already exists in the EEPROM.
1406	DNMXS_EEPROMError	The EEPROM has some errors.
1407	DNMXS_EEPROMNoData	The EEPROM is empty.

Table 4.2.6 Interpretation of the return code (CAN port Error) 1/1

Return Code	CAN chip Error	Comment
1500	DNMXS_SendCANMsgError	The CAN port can't send message.
1501	DNMXS_CANConfigError	The CAN port can't be configured.
1502	DNMXS_CANRxBufferError	The receive buffer has errors in the CAN port.
1503	DNMXS_CANInstallIRQError	There are errors when installing the interrupt resource.

Table 4.2.7 Interpretation of the return code (Functional Error) 1/3

Return Code	Functional Error	Comment
1600	DNMXS_OnlineError	**[Note1]**
1700	DNMXS_ConnectionTypeNotSupport	The connection type is not a valid I/O connection.
1800	DNMXS_BitStrobeAlreadyExistInEEP	The Bit-Strobe information already exists in the EEPROM.
1900	DNMXS_PollAlreadyExistInEEP	The Poll information already exists in the EEPROM.
2000	DNMXS_COSAlreadyExistInEEP	The COS information already exists in the EEPROM.
2100	DNMXS_CyclicAlreadyExistInEEP	The Cyclic information already exists in the EEPROM.
2200	DNMXS_PollOutputDataLenError	The output length of the Poll connection doesn't match the device's output length.
2300	DNMXS_COSOutputDataLenError	The output length of the COS connection doesn't match the device's output length.
2400	DNMXS_CyclicOutputDataLenError	The output length of the Cyclic connection doesn't match the device's output length.

[Note1]: The master MAC ID collides with other slave device in the DeviceNet network.

Table 4.2.8 Interpretation of the return code (Functional Error) 2/3

Return Code	Functional Error	Comment
2500	DNMXS_ExplicitUnConnect	The Explicit connection has not connected.
2501	DNMXS_PollUnConnect	The Poll connection has not connected.
2502	DNMXS_BitStrobeUnConnect	The Bit-Strobe connection has not connected.
2503	DNMXS_COSUnConnect	The COS connection has not connected.
2504	DNMXS_CyclicUnConnect	The Cyclic connection has not connected.

Table 4.2.9 Interpretation of the return code (Functional Error) 3/3

Return Code	Functional Error	Comment
2600	DNMXS_BusError	The CAN bus has some errors.
2601	DNMXS_MACIDDuplicate	There is the same MAC ID in the ScanList.
2700	DNMXS_BreakOptionError	The option is illegal when the communication is broken.
2800	DNMXS_WithoutPoll	The device doesn't support Poll connection.
2801	DNMXS_WithoutBitStrobe	The device doesn't support Bit -Strobe connection.
2802	DNMXS_WithoutCOS_Cyclic	The device doesn't support COS or Cyclic connection.
2900	DNMXS_DevicesError	There are slave devices which have some errors.

Table 4.2.10 Interpretation of the return code (Connection Status)1/3

Return Code	Connection Status	Comment
0	Status_Initial	Initial status
1	Status_ExplicitEstablishOK	Allocate Explicit connection successfully
2	Status_ExplicitEstablishing	Master is allocating Explicit connection
3	Status_ExplicitUnEstablish	Explicit connection doesn't be allocated.
4	Status_TimeoutActionUnSet	Explicit connection doesn't set Watchdog Timeout Action.
5	Status_TimeoutActionSetOK	Explicit connection has set Watchdog Timeout Action.
6	Status_TimeoutActionSetting	Explicit connection is setting Watchdog Timeout Action.
7	Status_IOReady	Ready to allocate I/O connection
10	Status_BitStrobeEstablishOK	Allocate Bit-Strobe connection successfully.
11	Status_BitStrobeEstablishing	Master is allocating Bit-Strobe Connection.
12	Status_BitStrobeUnEstablish	Bit-Strobe connection doesn't be allocated.
13	Status_BitStrobeGetingProducedSize	Bit-Strobe connection is getting Produced Connection Size
14	Status_BitStrobeGetProducedSizeOK	Bit-Strobe connection gets Produced Connection Size successfully.
15	Status_BitStrobeEPRUnSet	Bit-Strobe connection doesn't set Expected Packet Rate.
16	Status_BitStrobeEPRSetOK	Bit-Strobe connection has set Expected Packet Rate.
17	Status_BitStrobeEPRSetting	Bit-Strobe connection is setting Expected Packet Rate.

Table 4.2.11 Interpretation of the return code (Connection Status)2/3

Return Code	Connection Status	Comment
20	Status_PollEstablishOK	Allocate Poll connection successfully
21	Status_PollEstablishing	Master is allocating Poll connection.
22	Status_PollUnEstablish	Poll connection doesn't be allocated.
23	Status_PollGetingProducedSize	Poll connection is getting Produced Connection Size.
24	Status_PollGetProducedSizeOK	Poll connection gets Produced Connection Size successfully.
25	Status_PollGetingConsumedSize	Poll connection is getting Consumed Connection Size
26	Status_PollGetConsumedSizeOK	Poll connection gets Consumed Connection Size successfully.
27	Status_PollEPRUnSet	Poll connection doesn't set Expected Packet Rate.
28	Status_PollEPRSetOK	Poll connection has set Expected Packet Rate successfully.
29	Status_PollEPRSetting	Poll connection is setting Expected Packet Rate.

Table 4.2.12 Interpretation of the return code (Connection Status)3/3

Return Code	Connection Status	Comment
30	Status_COS_CyclicEstablishOK	Allocate COS_Cyclic Connection successfully.
31	Status_COS_CyclicEstablishing	Master is allocating COS_Cyclic Connection
32	Status_COS_CyclicUnEstablish	COS_Cyclic Connection doesn't be allocated.
33	Status_COS_CyclicGetingProducedSize	COS_Cyclic connection is getting COS_Cyclic Produced Connection Size
34	Status_COS_CyclicGetProducedSizeOK	COS_Cyclic connection gets Produced Connection Size successfully.
35	Status_COS_CyclicGetingConsumedSize	COS_Cyclic connection is getting Consumed Connection Size
36	Status_COS_CyclicGetConsumedSizeOK	COS_Cyclic connection gets Consumed Connection Size successfully.
37	Status_COS_CyclicEPRUnSet	COS_Cyclic connection doesn't set Expected Packet Rate.
38	Status_COS_CyclicEPRSetOK	COS_Cyclic connection has set Expected Packet Rate successfully.
39	Status_COS_CyclicEPRSetting	COS_Cyclic connection is setting Expected Packet Rate.
40	Status_COS_CyclicACKErr	COS_Cyclic ACK replies error.
41	Status_COS_CyclicACKTMO	COS_Cyclic ACK does not reply.

Table 4.2.13 Interpretation of the return code (Connection Result)1/5

Return Code	Connection Result	Comment
0	Result_None	None
1	Result_AllocateResOK	Allocation replies OK.
2	Result_AllocateResERR	Allocation replies error.
3	Result_AllocateResTMO	Allocation replies timeout.
4	Result_SetActionResOK	Setting Watchdog Timeout Action is OK.
5	Result_SetActionResERR	Setting Watchdog Timeout Action responses error.
6	Result_SetActionResTMO	Setting Watchdog Timeout Action responses timeout.
7	Result_ReleaseResOK	Releasing connection responses OK.
8	Result_ReleaseResERR	Releasing connection responses error.
9	Result_ReleaseResTMO	Releasing connection responses timeout.
10	Result_GetAttrResOK	Getting Attribute is OK.
11	Result_GetAttrResERR	Getting Attribute is error.
12	Result_GetAttrResTMO	Getting Attribute is timeout.
13	Result_SetAttrResOK	Setting Attribute is OK.
14	Result_SetAttrResERR	Setting Attribute is error.
15	Result_SetAttrResTMO	Setting Attribute is timeout.
16	Result_SetAttrACKTMO	Setting Attribute ACK is timeout.

Table 4.2.14 Interpretation of the return code (Connection Result)2/5

Return Code	Connection Result	Comment
20	Result_BitStrobeEPRResOK	Setting the Expected Packet Rate of the Bit-Strobe connection replies OK.
21	Result_BitStrobeEPRResERR	Setting the Expected Packet Rate of the Bit-Strobe connection replies error.
22	Result_BitStrobeEPRResTMO	Setting the Expected Packet Rate of the Bit-Strobe connection does not reply.
23	Result_BitStrobeIOResOK	The Bit-Strobe I/O connection of the slave device replies OK.
24	Result_BitStrobeIOResERR	The Bit-Strobe I/O connection of the slave device replies error.
25	Result_BitStrobeIOResTMO	The Bit-Strobe I/O connection of the slave device does not reply.
26	Result_BitStrobeProducedResErr	Getting the Produced Size of the Bit-Strobe connection replies error.
27	Result_BitStrobeProducedResTMO	Getting the Produced Size of the Bit-Strobe connection does not reply.
28	Result_BitStrobeProducedSizeError	The input data length parameter of the Bit-Strobe connection which has been set by the users does not match with the length of the slave device.

Table 4.2.15 Interpretation of the return code (Connection Result)3/5

Return Code	Connection Result	Comment
30	Result_PollEPRResOK	Setting the Expected Packet Rate of the Poll connection replies OK.
31	Result_PollEPRResERR	Setting the Expected Packet Rate of the Poll connection replies error.
32	Result_PollEPRResTMO	Setting the Expected Packet Rate of the Poll connection does not reply.
33	Result_PollIOResOK	The Poll I/O connection of the slave device replies OK.
34	Result_PollIOResERR	The Poll I/O connection of the slave device replies error.
35	Result_PollIOResTMO	The Poll I/O connection of the slave device does not reply.
36	Result_PollProducedResErr	Getting the Produced Size of the Poll connection replies error.
37	Result_PollProducedResTMO	Getting the Produced Size of the Poll connection does not reply.
38	Result_PollProducedSizeError	The input data length parameter of the Poll connection which has been set by the users does not match with the length of the slave device.
39	Result_PollConsumedResErr	Getting the Consumed Size of the Poll connection replies error.
40	Result_PollConsumedResTMO	Getting the Consumed Size of the Poll connection does not reply.
41	Result_PollConsumedSizeError	The output data length parameter of the Poll connection which has been set by the users does not match with the length of the slave device.

Table 4.2.16 Interpretation of the return code (Connection Result)4/5

Return Code	Connection Result	Comment
53	Result_COS_CyclicEPRResOK	Setting the Expected Packet Rate of COS_Cyclic connection replies OK.
54	Result_COS_CyclicEPRResERR	Setting the Expected Packet Rate of COS_Cyclic connection replies error.
55	Result_COS_CyclicEPRResTMO	Setting the Expected Packet Rate of COS_Cyclic connection does not reply.
56	Result_COS_CyclicIOResOK	The COS_Cyclic I/O connection replies OK.
57	Result_COS_CyclicIOResERR	The COS_Cyclic I/O connection replies error.
58	Result_COS_CyclicIOResTMO	The COS_Cyclic I/O connection does not reply.
59	Result_COS_CyclicACKResOK	The COS_Cyclic ACK replies OK.
60	Result_COS_CyclicACKResERR	The COS_Cyclic ACK replies error.
61	Result_COS_CyclicACKResTMO	The COS_Cyclic ACK does not reply.

Table 4.2.17 Interpretation of the return code (Connection Result)5/5

Return Code	Connection Result	Comment
62	Result_COS_CyclicProducedResErr	Getting the Produced Size of the COS_Cyclic connection replies error.
63	Result_COS_CyclicProducedResTMO	Getting the Produced Size of the COS_Cyclic connection does not reply.
64	Result_COS_CyclicProducedSizeError	The input data length parameter of the COS_Cyclic connection which has been set by the users does not match with the length of the slave device.
65	Result_COS_CyclicConsumedResErr	Getting the Consumed Size of the COS_Cyclic connection replies error.
66	Result_COS_CyclicConsumedResTMO	Getting the Consumed Size of the COS_Cyclic connection does not reply.
67	Result_COS_CyclicConsumedSizeError	The output data length parameter of the COS_Cyclic connection which has been set by the users does not match with the length of the slave device.
70	Result_ModuleNoResponse	Slave Module doesn't response

4.3 Flow Diagram for Auto-check UCMM Architecture

PISO-DNM100 supports Group 2 only Server and UCMM functions to communication with slave devices. The users will not see any UCMM functions. How to communicate with UCMM slave devices? Actually, the firmware inside PISO-DNM100 will check the type of the slave devices automatically. The users need not to know what the type of the remote device is. If the slave device is UCMM capable, the PISO-DNM100 will communicate with it by the UCMM mode automatically. Relatively, if the slave device is UCMM incapable, the PISO-DNM100 will communicate with it by the Predefined Master-Slave Connection Set automatically. The PISO-DNM100 will detect the capability of the slave device and change to the corresponding mode automatically. The figure 4.2 shows how the firmware inside PISO-DNM100 works.

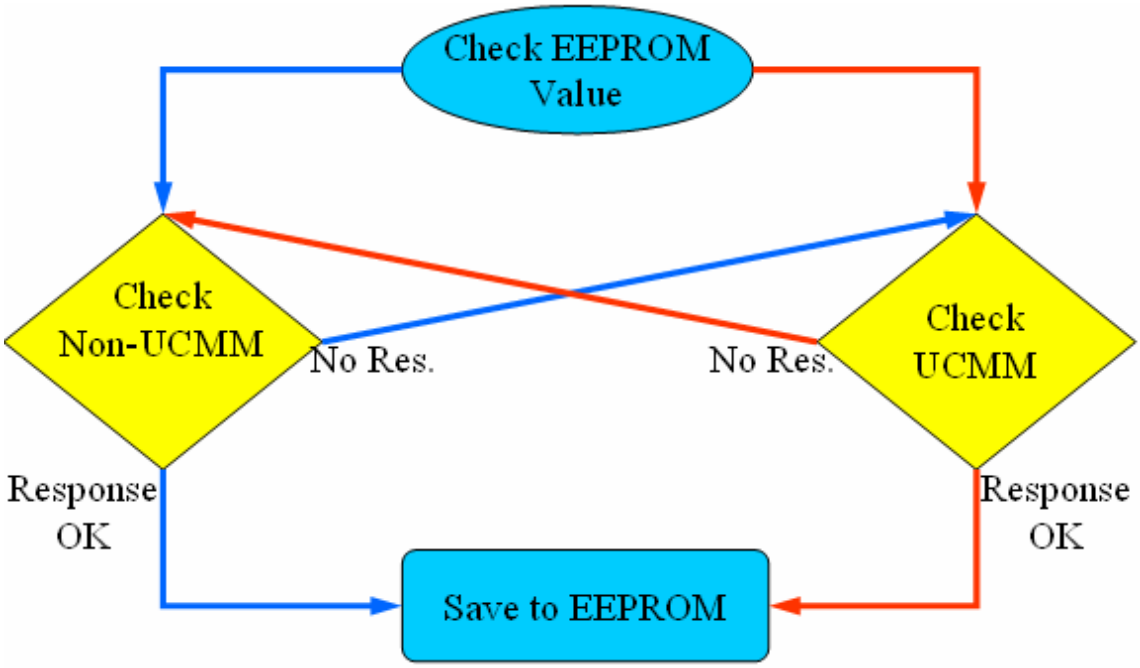


Figure 4.2 Auto-check UCMM Architecture

4.4 Function Description

4.4.1 DNM100_GetBoardInf

- **Description:**

  This function is used to obtain the driver information of PISO-DNM100 board.

- **Syntax:**

```
DWORD DNM100_GetBoardInf (BYTE BoardNo, DWORD *dwVID,
                          DWORD *dwDID, DWORD *dwSVID,
                          DWORD *dwSDID, DWORD *dwSAuxID,
                          DWORD *dwIrqNo)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  dwVID: [output] The address of a variable which is used to receive the vendor ID.
  dwDID: [output] The address of a variable used to receive device ID.
  dwSVID: [output] The address of a variable applied to receive sub-vendor ID.
  dwSDID: [output] The address of a variable applied to receive sub-device ID.
  dwSAuxID: [output] The address of a variable used to receive sub-auxiliary ID.
  dwIrqNo: [output] The address of a variable used to receive logical interrupt number.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.

4.4.2 DNM100_TotalDNM100Board

- **Description:**

  The function can get the count of total PISO-DNM100 boards in the user's PC.

- **Syntax:**

  DWORD DNM100_TotalDNM100Board (BYTE *TotalBoards,
  BYTE *BoardIDList);

- **Parameter:**

  TotalBoards: [output] The count of total board.
  BoardIDList: [output] The list of all DIP-Switch No. in each board.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.

4.4.3 DNM100_ActiveBoard

- **Description:**

  The function is used to activate PISO-DNM100-D/T. It must be called once before using the other functions of PISO-DNM100-D/T APIs.

- **Syntax:**

  DWORD DNM100_ActiveBoard (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.

4.4.4 DNM100_CloseBoard

- **Description:**

  The function is used to stop and close the kernel driver and release the device resource from computer device resource. This method must be called once before exiting the user's application program.

- **Syntax:**

  DWORD DNM100_CloseBoard (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK

  DNM100_DriverError (10001): Kernel driver is not opened.

  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.

4.4.5 DNM100_GetDLLVersion

- **Description:**

  The function can obtain the version information of DNM100.dll driver.

- **Syntax:**

  DWORD DNM100_GetDLLVersion (void)

- **Parameter:**

  None

- **Return:**

  The DLL version information. For example: If 100(hex) is return, it means driver version is 1.00.

4.4.6 DNM100_GetFirmwareVersion

- **Description:**

  The function can obtain the version information of the firmware inside PISO-DNM100.

- **Syntax:**

  DWORD DNM100_GetFirmwareVersion (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  The firmware version information. For example: If 100(hex) is return, it means firmware version is 1.00.

- **Error Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.

4.4.7 DNM100_ResetFirmware

- **Description:**

  The function is used to reset the PISO-DNM100 firmware. When the users have changed the baud rate of CAN bus or changed the Master's MAC ID, the function must be called to make the change enable. After calling this function, the users should wait for 1 or 2 seconds to make the firmware boot up completely.

- **Syntax:**

  DWORD DNM100_ResetFirmware (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  DNM100_ModeError (10027): This board is in download mode, and can't be changed to firmware mode.

4.4.8 DNM100_CheckFirmwareMode

- **Description:**

  The function is used to obtain the specified PISO-DNM100-D/T if it is in download mode or firmware mode.

- **Syntax:**

  DWORD DNM100_CheckFirmwareMode (BYTE BoardNo, BYTE *Mode)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

  Mode: [output] The address of a variable used to get the PISO-DNM100-D/T mode. If this value is 0, it indicates that the PISO-DNM100-D/T is in download mode. If 1, it is in firmware mode. When PISO-DNM100-D/T is in download mode, it can only update the firmware and the firmware will not work at the same time.

- **Return:**

  DNM100_NoError (0): OK

  DNM100_DriverError (10001): Kernel driver is not opened.

  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.

  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.

  DNM100_InitError (10007): The PISO-DNM100 replies erroneously.

  DNM100_TimeOut (10023): The PISO-DNM100 has no response.

  DNM100_ModeError (10027): This board is in download mode, and can't be changed to firmware mode.

4.4.9 DNM100_GetMasterMACID

- **Description:**

  The function can get the MAC ID of the DeviceNet master (PISO-DNM100).

- **Syntax:**

  DWORD DNM100_GetMasterMACID (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  The Master's MAC ID. (0 ~ 63)

- **Error Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.10 DNM100_SetMasterMACID

- **Description:**

  The function can set the MAC ID of the DeviceNet master (PISO-DNM100). After calling this function, the users must call DNM100_ResetFirmware to make the change enabled. It will save the information in the EEPROM in PISO-DNM100.

- **Syntax:**

  DWORD DNM100_SetMasterMACID (BYTE BoardNo, BYTE MasterMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  MasterMACID: [input] The DeviceNet master's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.11 DNM100_GetBaudRate

- **Description:**

  This function can help you to get the baud rate information of PISO-DNM100.

- **Syntax:**

  DWORD DNM100_GetBaudRate (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  The baud rate information in the PISO-DNM100.
  If the value is 0, the baud rate is 125Kbps.
  If the value is 1, the baud rate is 250Kbps.
  If the value is 2, the baud rate is 500Kbps.

- **Error Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.12 DNM100_SetBaudRate

- **Description:**

  This function can set the DeviceNet baud rate of the PISO-DNM100. After calling this function, you must call DNM100_ResetFirmware to reset the firmware to make change enabled.

- **Syntax:**

  DWORD DNM100_SetBaudRate (BYTE BoardNo, BYTE BaudRate)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  BaudRate: [input] The new baud rate value.
  0 : 125K bps
  1 : 250K bps
  2 : 500K bps

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.13 DNM100_GetMasterStatus

- **Description:**

  The function is used to obtain the firmware status inside PISO-DNM100. The users can call this function to make sure that the DeviceNet master is online successfully.

- **Syntax:**

  DWORD DNM100_GetMasterStatus (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.14 DNM100_StartDevice

- **Description:**

  This function is used to start to communicate with the specific device that the users applying.

- **Syntax:**

  DWORD DNM100_StartDevice (BYTE BoardNo,BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.15 DNM100_StopDevice

- **Description:**

  This function is used to stop to communicate with the destination device that the users appointed.

- **Syntax:**

  DWORD DNM100_StopDevice (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.16 DNM100_StartAllDevice

- **Description:**

  This function is used to start to communicate with all slave devices in ScanList.

- **Syntax:**

  DWORD DNM100_StartAllDevice (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.17 DNM100_StopAllDevice

- **Description:**

  This function is used to stop to communicate with all destination devices in ScanList.

- **Syntax:**

  DWORD DNM100_StopAllDevice (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.18 DNM100_AddDevice

- **Description:**

  This function can add the slave devices into the ScanList of the PISO-DNM100 and save the information into the EEPROM. Before communicating with any slave devices, the users should call this function to add these devices.

- **Syntax:**

```
DWORD DNM100_AddDevice (BYTE BoardNo,
                        BYTE DesMACID,
                        WORD Explicit_EPR)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  Explicit_EPR: [input] The Expected Packet Rate. (Usually is 2500).

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.19 DNM100_RemoveDevice

- **Description:**

  This function is used for removing the specified slave device from the ScanList in the PISO-DNM100. And the information of the device in EEPROM is erased at the same time.

- **Syntax:**

  DWORD DNM100_RemoveDevice (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.20 DNM100_GetExplicitStatus

- **Description:**

  The function is used to check the Explicit Message Connection Status, which connects to the specific slave. After you have started the device, you can execute this function to check the connection's status.

- **Syntax:**

  DWORD DNM100_GetExplicitStatus (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.10 ~ Table 4.2.12.

4.4.21 DNM100_GetExplicitResult

- **Description:**

  This function is used to check the Explicit Connection result with the specific slave device.

- **Syntax:**

  DWORD DNM100_GetExplicitResult (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.13 ~ Table 4.2.17.

4.4.22 DNM100_GetAttribute

- **Description:**

  This function is used to send the request command to retrieve the attribute value of the specific device's instance. Before calling this function, you must start the device. After calling this function, you should execute the "DNM100_GetAttributeValue" to get the response message returned from remote slave device.

- **Syntax:**

  DWORD DNM100_GetAttribute (BYTE BoardNo, BYTE DesMACID, BYTE ClassID, BYTE InstanceID, BYTE AttributeID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  ClassID: [input] The remote slave device's ClassID
  InstanceID: [input] The remote slave device's InstanceID
  AttributeID: [input] The remote slave device's AttributeID

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9.

4.4.23 DNM100_GetAttributeValue

- **Description:**

  This function is used to get the attribute value of the specific device's instance, which returned from the remote slave device. Before calling this function, the users should call DNM100_GetAttribute to send request command first.

- **Syntax:**

  DWORD DNM100_GetAttributeValue (BYTE BoardNo, BYTE DesMACID, WORD *DataLen, BYTE *DATA)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DataLen: [output] The length of the attribute value (in byte).
  DATA: [output] The attribute value that returned from the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9 and Table 4.2.13 ~ Table 4.2.17.

4.4.24 DNM100_SetAttribute

- **Description:**

  The method is used to set the attribute of the specific device's instance. Before calling this function, you must start the device. After calling this function, you should execute the "DNM100_SetAttributeResponse" to check the response message returned from the remote slave device.

- **Syntax:**

  DWORD DNM100_SetAttribute (BYTE BoardNo, BYTE DesMACID, BYTE ClassID, BYTE InstanceID, BYTE AttributeID, WORD DataLen, BYTE *DATA)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  ClassID: [input] The remote slave device's ClassID
  InstanceID: [input] The remote slave device's InstanceID
  AttributeID: [input] The remote slave device's AttributeID
  DataLen: [input] The length of the attribute value (in byte).
  DATA: [input] The attribute value that the users want to send.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.25 DNM100_SetAttributeResponse

- **Description:**

  This function is used to get the response value after executing the "DNM100_SetAttribute" function.

- **Syntax:**

  DWORD DNM100_SetAttributeResponse (BYTE BoardNo,
  BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9 and Table 4.2.13 ~ Table 4.2.17.

4.4.26 DNM100_RemoveIOConnection

- **Description:**

  The function is used to remove the I/O Connection configuration.

- **Syntax:**

  DWORD DNM100_RemoveIOConnection (BYTE BoardNo,
  BYTE DesMACID, BYTE Type)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  Type: [input] The I/O Connection type. (1 ~ 4).
  1 : Poll connection.
  2 : Bit-Strobe connection.
  3 : COS connection.
  4 : Cyclic connection.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.27 DNM100_SetBreakOption

- **Description:**

  This function will change the re-connection mechanism when the slave device is broken.

- **Syntax:**

  DWORD DNM100_SetBreakOption (BYTE BoardNo, BYTE DesMACID, BYTE Option)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  Option: [input] The option value when connection is broken.
  0x00 : No re-connection mechanism.
  0x01 : Re-Connect when I/O connection is broken. If the slave device doesn't response or response error. The master (PISO-DNM100) will re-connect it. If the device was shut down then power on, the master will not re-connect it.
  0x02 : Re-Connect when device was power on again.
  0x03 : The combination of the 0x01 and 0x02. This is the default value when the users add new devices into ScanList.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.28 DNM100_GetBreakOption

- **Description:**

  This function will get the option value when slave device connection is broken.

- **Syntax:**

  DWORD DNM100_GetBreakOption (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  The option value of the slave device.
  0x00 : No re-connection mechanism.
  0x01 : Re-Connect when I/O connection is broken. If the slave device doesn't response or response error. The master (PISO-DNM100) will re-connect it. If the device was shut down then power on, the master will not re-connect it.
  0x02 : Re-Connect when device was power on again.
  0x03 : The combination of the 0x01 and 0x02. This is the default value when the users add new device into ScanList.

- **Error Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.29 DNM100_ClearAllConfig

- **Description:**

  This function will clear all configurations in the EEPROM of the PISO-DNM100.

- **Syntax:**

  DWORD DNM100_ClearAllConfig (BYTE BoardNo)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.30 DNM100_GetLastDeviceError

- **Description:**

  This function will get the last error device in the network

- **Syntax:**

  DWORD DNM100_GetLastDeviceError (BYTE BoardNo,
  BYTE *Err_MACID,
  BYTE *Err_Type,
  BYTE *Err_Code)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  Err_MACID: [output] The error slave device's MAC ID. If there is no error device, the value is FF (hex).
  Err_Type: [output] The connection type of the error slave device. If there is no error device, the value is FF (hex).
  Err_Code: [output] The error code. If there is no error device, the value is FF (hex). Please refer to Table 4.2.13 ~ Table 4.2.17

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.31 DNM100_GetDeviceStatus

- **Description:**

  This function is to get the remote slave device's communication all error status.

- **Syntax:**

  DWORD DNM100_GetDeviceStatus (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.32 DNM100_DebugDevice

- **Description:**

  This function will try to connect to the remote device. Attention! This function will terminate the communication with appoint device. This function is usually used to check the wire connection during developing applications.

- **Syntax:**

  DWORD DNM100_DebugDevice (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.33 DNM100_GetDeviceInfo

- **Description:**

  This function will get the specific device's information in the ScanList.

- **Syntax:**

  DWORD DNM100_GetDeviceInfo (BYTE BoardNo, BYTE DesMACID,
  WORD *ListCount,
  BYTE *ConnectionTypeList,
  WORD *InputDataLenList,
  WORD *OutputDataLenList,
  WORD *EPRList)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  ListCount: [output] The data count of all the information.
  ConnectionTypeList: [output] The connection type of the slave device.
  InputDataLenList: [output] The input data length of the slave device.
  OutputDataLenList: [output] The output data length of the slave device.
  EPRList: [output] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.34 DNM100_GetDevicePollInfo

- **Description:**

  This function will try to connect to the remote device by the Poll connection. Attention! This function will terminate the communication with the appoint device. This function is usually used to check the wire connection and check the Poll configuration during developing applications.

- **Syntax:**

```
DWORD DNM100_GetDevicePollInfo (BYTE BoardNo,
                                BYTE DesMACID,
                                WORD *PollInputLen,
                                WORD *PollOutputLen)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  PollInputLen: [output] The input data length of the slave device.
  PollOutputLen: [output] The output data length of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.35 DNM100_GetDeviceBitStrobeInfo

- **Description:**

  This function will try to connect to the remote device by Bit-Strobe connection. Attention! This function will terminate the communication with the appoint device. This function is usually used to check the wire connection and check the Bit-Strobe configuration during developing applications.

- **Syntax:**

```
DWORD DNM100_GetDeviceBitStrobeInfo (BYTE BoardNo,
                                     BYTE DesMACID,
                                     WORD *BitStrobeInputLen)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  BitStrobeInputLen: [output] The input data length of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.36 DNM100_GetDeviceCOSInfo

- **Description:**

  This function will try to connect to the remote device by COS connection. Attention! This function will terminate the communication with the appoint device. This function is usually used to check the wire connection and check the COS configuration during developing applications.

- **Syntax:**

```
DWORD DNM100_GetDeviceCOSInfo (BYTE BoardNo,
                               BYTE DesMACID,
                               WORD *COSInputLen,
                               WORD *COSOutputLen)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  COSInputLen: [output] The input data length of the slave device.
  COSOutputLen: [output] The output data length of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.37 DNM100_GetDeviceCyclicInfo

- **Description:**

  This function will try to connect to the remote device by Cyclic connection. Attention! This function will terminate the communication with the appoint device. This function is usually used to check the wire connection and check the Cyclic configuration during developing applications.

- **Syntax:**

```
DWORD DNM100_GetDeviceCyclicInfo (BYTE BoardNo,
                                  BYTE DesMACID,
                                  WORD *CyclicInputLen,
                                  WORD *CyclicOutputLen)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  CyclicInputLen: [output] The input data length of the slave device.
  CyclicOutputLen: [output] The output data length of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.38 DNM100_AutoScanDevice

- **Description:**

  This function is used to retrieve all devices in DeviceNet network. Attention! This function will terminate all communications with remote devices. This function is usually used for developing or debugging applications.

- **Syntax:**

```
DWORD DNM100_AutoScanDevice (BYTE BoardNo,
                             WORD *TotalDevices,
                             BYTE *DesMACID,
                             BYTE *Type,
                             WORD *DeviceInputLen,
                             WORD *DeviceOutputLen)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  TotalDevices: [output] The data count of all the information.
  DestMACID: [output] The MAC ID of all the slave devices.
  Type: [output] The connection type of all the slave devices.
  DeviceInputLen: [output] The input data length of all the slave devices.
  DeviceOutputLen: [output] The output data length of all the slave devices.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.39 DNM100_GetScanList

- **Description:**

  This function will get the ScanList data in the PISO-DNM100.

- **Syntax:**

```
DWORD DNM100_GetScanList (BYTE BoardNo, WORD *TotalDevices,
                          BYTE *DesMACIDList,
                          BYTE *ConnectionTypeList,
                          WORD *InputDataLenList,
                          WORD *OutputDataLenList,
                          WORD *EPR_List)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  TotalDevices: [output] The data count of all the information.
  DestMACIDList: [output] The MAC ID of all the slave devices in the ScanList.
  ConnectionTypeList: [output] The connection type of all the slave devices in the ScanList.
  InputDataLenList: [output] The input data length of all the slave devices in the ScanList.
  OutputDataLenList: [output] The output data length of all the slave devices in the ScanList.
  EPR_List: [output] The EPR value of all the slave devices in the ScanList.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.40 DNM100_ConfigPoll

- **Description:**

  This method is used to configure the Poll connection of the specific MAC ID device. The PISO-DNM100 can get/set the data via the connection, which connects to the specific slave, according to the produced / consumed connection path of this slave device. This configuration data will be saved into EEPROM on the PISO-DNM100.

- **Syntax:**

```
DWORD DNM100_ConfigPoll (BYTE BoardNo, BYTE DesMACID,
                         WORD DeviceInputLen,
                         WORD DeviceOutputLen,
                         WORD EPR)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  DeviceOutputLen: [input] The output data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.41 DNM100_UpdatePollConfig

- **Description:**

  The function is to update the original Poll information. This configuration data will be updated into the EEPROM of the PISO-DNM100.

- **Syntax:**

  DWORD DNM100_UpdatePollConfig (BYTE BoardNo, BYTE DesMACID,
  WORD DeviceInputLen,
  WORD DeviceOutputLen,
  WORD EPR)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  DeviceOutputLen: [input] The output data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.42 DNM100_ReadPollInputData

- **Description:**

  This function is to get the data according with the produced connection path of the specific MAC ID device via the Poll connection.

- **Syntax:**

```
DWORD DNM100_ReadPollInputData (BYTE BoardNo,
                                BYTE DesMACID,
                                WORD *IOLen,
                                BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [output] The length of the I/O data (In byte).
  IODATA: [output] The remote I/O data.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.43 DNM100_WritePollOutputData

- **Description:**

  The function will set the data according with the consumed connection path of the specific MAC ID device via the Poll connection.

- **Syntax:**

```
DWORD DNM100_WritePollOutputData (BYTE BoardNo,
                                  BYTE DesMACID,
                                  WORD IOLen,
                                  BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [input] The length of the I/O data (In byte).
  IODATA: [input] The remote I/O data which are defined by users.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.44 DNM100_GetPollStatus

- **Description:**

  The function is used to get the Poll connection status value, which connects to the specific MAC ID device.

- **Syntax:**

  DWORD DNM100_GetPollStatus (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.45 DNM100_GetPollResult

- **Description:**

  This function is used to check the error status of the remote slave device via Poll I/O Connection.

- **Syntax:**

  DWORD DNM100_GetPollResult (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9 and Table 4.2.13 ~ Table 4.2.17

4.4.46 DNM100_CheckPollConnectionStatus

- **Description:**

  The function is to check whether the Poll connection has normal status or not. The users can call this function to check the connection status cyclically.

- **Syntax:**

  DWORD DNM100_CheckPollConnectionStatus (BYTE BoardNo,
  BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.47 DNM100_ConfigBitStrobe

- **Description:**

  The function is to configure the Bit-Strobe connection of the specific MAC ID device. The PISO-DNM100 will get the data of all slave devices which support Bit-Strobe connection in every period of time via this connection. The period of the time is 1 second. This configuration data will be saved into EEPROM on the PISO-DNM100.

- **Syntax:**

  DWORD DNM100_ConfigBitStrobe (BYTE BoardNo, BYTE DesMACID, WORD DeviceInputLen, WORD EPR)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.48 DNM100_UpdateBitStrobeConfig

- **Description:**

  The function is to update the original Bit-Strobe information. This configuration data will be updated into the EEPROM of the PISO-DNM100.

- **Syntax:**

```
DWORD DNM100_UpdateBitStrobeConfig (BYTE BoardNo,
                                    BYTE DesMACID,
                                    WORD DeviceInputLen,
                                    WORD EPR)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.49 DNM100_ReadBitStrobeInputData

- **Description:**

  The function is used to get the data according with the produced connection path of the specific slave device via the Bit-strobe connection.

- **Syntax:**

```
DWORD DNM100_ReadBitStrobeInputData (BYTE BoardNo,
                                     BYTE DesMACID,
                                     WORD *IOLen,
                                     BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [output] The length of the I/O data (In byte).
  IODATA: [output] The remote I/O data.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.50 DNM100_GetBitStrobeStatus

- **Description:**

  Obtain the status value of the Bit-Strobe connection between the PISO-DNM100 and the specific slave device.

- **Syntax:**

  DWORD DNM100_GetBitStrobeStatus (BYTE BoardNo,
  BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.51 DNM100_GetBitStrobeResult

- **Description:**

  This function is used to check the error status of the remote slave device via Bit-Strobe I/O Connection.

- **Syntax:**

  DWORD DNM100_GetBitStrobeResult (BYTE BoardNo,
  BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9 and Table 4.2.13 ~ Table 4.2.17

4.4.52 DNM100_CheckBitStrobeConnectionStatus

- **Description:**

  The function is to check whether the Bit-Strobe connection has normal status or not. The users can call this function to check the connection status cyclically.

- **Syntax:**

  DWORD DNM100_CheckBitStrobeConnectionStatus (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.53 DNM100_ConfigCOS

- **Description:**

  This function is used to configure the Change_Of_State connection of the specific device. After configuring the connection, the specific slave device would send out the I/O data when the I/O status changed. This configuration data will be saved into EEPROM on the PISO-DNM100.

- **Syntax:**

  DWORD DNM100_ConfigCOS (BYTE BoardNo, BYTE DesMACID,
  WORD DeviceInputLen,
  WORD DeviceOutputLen,
  WORD EPR)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  DeviceOutputLen: [input] The output data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.54 DNM100_UpdateCOSConfig

- **Description:**

  The function is to update the original COS information. This configuration data will be updated into the EEPROM of the PISO-DNM100.

- **Syntax:**

```
DWORD DNM100_UpdateCOSConfig (BYTE BoardNo,
                              BYTE DesMACID,
                              WORD DeviceInputLen,
                              WORD DeviceOutputLen,
                              WORD EPR)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  DeviceOutputLen: [input] The output data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.55 DNM100_ReadCOSInputData

- **Description:**

  This method is used to read the data of the specific slave device via the COS connection.

- **Syntax:**

```
DWORD DNM100_ReadCOSInputData (BYTE BoardNo,
                               BYTE DesMACID,
                               WORD *IOLen,
                               BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [output] The length of the I/O data (In byte).
  IODATA: [output] The remote I/O data.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.56 DNM100_WriteCOSOutputData

- **Description:**

  This function is used to set the data according with the consumed connection path of the specific slave device via the COS connection.

- **Syntax:**

```
DWORD DNM100_WriteCOSOutputData (BYTE BoardNo,
                                 BYTE DesMACID,
                                 WORD IOLen,
                                 BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [input] The length of the I/O data (In byte).
  IODATA: [input] The remote I/O data which are defined by users.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.57 DNM100_GetCOSStatus

- **Description:**

  This function is used to get the status value of the COS connection which is connected to the MAC ID slave device.

- **Syntax:**

  DWORD DNM100_GetCOSStatus (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.58 DNM100_GetCOSResult

- **Description:**

  This function is used to check the error status of the remote slave device via COS I/O Connection.

- **Syntax:**

  DWORD DNM100_GetCOSResult (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9 andTable 4.2.13 ~ Table 4.2.17

4.4.59 DNM100_CheckCOSConnectionStatus

- **Description:**

  The function is to check whether the COS connection has normal status or not. The users can call this function to check the connection status cyclically.

- **Syntax:**

  DWORD DNM100_CheckCOSConnectionStatus (BYTE BoardNo,
  BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.60 DNM100_ConfigCyclic

- **Description:**

  This function is used to configure the Cyclic connection. This connection will get the data in every period of time. The period of the time is the value of the EPR (in millisecond) parameters. This configuration data will be saved into EEPROM on the PISO-DNM100.

- **Syntax:**

```
DWORD DNM100_ConfigCyclic (BYTE BoardNo, BYTE DesMACID,
                           WORD DeviceInputLen,
                           WORD DeviceOutputLen,
                           WORD EPR)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  DeviceOutputLen: [input] The output data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.61 DNM100_UpdateCyclicConfig

- **Description:**

  The function is to update the original Cyclic information. This configuration data will be update into EEPROM of the PISO-DNM100.

- **Syntax:**

```
DWORD DNM100_UpdateCyclicConfig (BYTE BoardNo,
                                 BYTE DesMACID,
                                 WORD DeviceInputLen,
                                 WORD DeviceOutputLen,
                                 WORD EPR)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  DeviceInputLen: [input] The input data length of the slave device.
  DeviceOutputLen: [input] The output data length of the slave device.
  EPR: [input] The EPR value of the slave device.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.62 DNM100_ReadCyclicInputData

- **Description:**

  This function is used to get the data according with the produced connection path of the MAC ID slave device via the cyclic connection.

- **Syntax:**

```
DWORD DNM100_ReadCyclicInputData (BYTE BoardNo,
                                  BYTE DesMACID,
                                  WORD *IOLen,
                                  BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [output] The length of the I/O data (In byte).
  IODATA: [output] The remote I/O data.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.63 DNM100_WriteCyclicOutputData

- **Description:**

  This method is used to set the data according with the Cyclic's consumed connection path of the output channel of the device.

- **Syntax:**

```
DWORD DNM100_WriteCyclicOutputData (BYTE BoardNo,
                                    BYTE DesMACID,
                                    WORD IOLen,
                                    BYTE *IODATA)
```

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)
  IOLen: [input] The length of the I/O data (In byte).
  IODATA: [input] The remote I/O data which are defined by users.

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9

4.4.64 DNM100_GetCyclicStatus

- **Description:**

  This function is used to obtain the status value of the Cyclic connection which is connected to the MAC ID slave device.

- **Syntax:**

  DWORD DNM100_GetCyclicStatus (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

4.4.65 DNM100_GetCyclicResult

- **Description:**

  This function is used to check the error status of the remote slave device via Cyclic I/O Connection.

- **Syntax:**

  DWORD DNM100_GetCyclicResult (BYTE BoardNo, BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.9 and Table 4.2.13 ~ Table 4.2.17

4.4.66 DNM100_CheckCyclicConnectionStatus

- **Description:**

  The function is to check whether the COS connection has normal status or not. The users can call this function to check the connection status cyclically.

- **Syntax:**

  DWORD DNM100_CheckCyclicConnectionStatus (BYTE BoardNo,
  BYTE DesMACID)

- **Parameter:**

  BoardNo: [input] PISO-DNM100 board number (0~15)
  DestMACID: [input] The remote slave device's MAC ID (0~63)

- **Return:**

  DNM100_NoError (0): OK
  DNM100_DriverError (10001): Kernel driver is not opened.
  DNM100_ActiveBoardError (10002): This board can not be activated or kernel driver can not be found.
  DNM100_BoardNumberError (10003): BoardNo exceeds the current scanned total board number.
  DNM100_TimeOut (10023): The PISO-DNM100 has no response.
  Others: Please refer to Table 4.2.2 ~ Table 4.2.12

5. Demo Programs for Windows

All of demo programs will not work normally if PISO-DNM100 driver would not be installed correctly. During the installation process of the driver, the install-shields will register the correct kernel driver to the operation system and copy the DLL driver and demo programs to the correct position based on the driver software package you have selected (Win98,Me,NT,win2000,XP). After completing the driver installation, the related demo programs, development library and declaration header files for different development environments are installed in the system as follows.

The PISO-DNM100's root directory is C:\ICPDAS\PISO-DNM100

|--\DLL	The DLL driver for the user's application
|--\Driver	The window driver of the PISO-DNM100
|--\Manual	The user manual of the PISO-DNM100
|--\Utility	The DeviceNet Master utility
|--\Demo	Demo program
|--\Demo\BCB 6	Demos for Borland C++ Builder 6
|--\Demo\VC++ 6	Demos for Visual C++ 6

5.1 A brief introduction to the demo programs

VC_Demo1 : Demonstrate the basic functions to communicate with the remote slave device. The demo program will lead you step by step to complete the setting and communication.

VC_Demo2 : Demonstrate the scan function to scan all the remote slave devices in the same DeviceNet network. The demo program will show you all the slave devices and their I/O connection type.

BCB_Demo1 : Demonstrate the scan function and add/remove function to configure the information of the remote slave device.

BCB_Demo2 : Demonstrate the I/O functions to access the I/O data of the remote slave device. The demo program will show you the input value of the remote device and let you send out the data to the output pins of the remote slave devices.

5.2 Wire Connection of the CAN bus

Before starting the demos, the users should have at least one slave device. Here show the users how to connect the master and slave devices by CAN bus. The slave devices should be connected to form the serial type which is shown as Figure 5.1

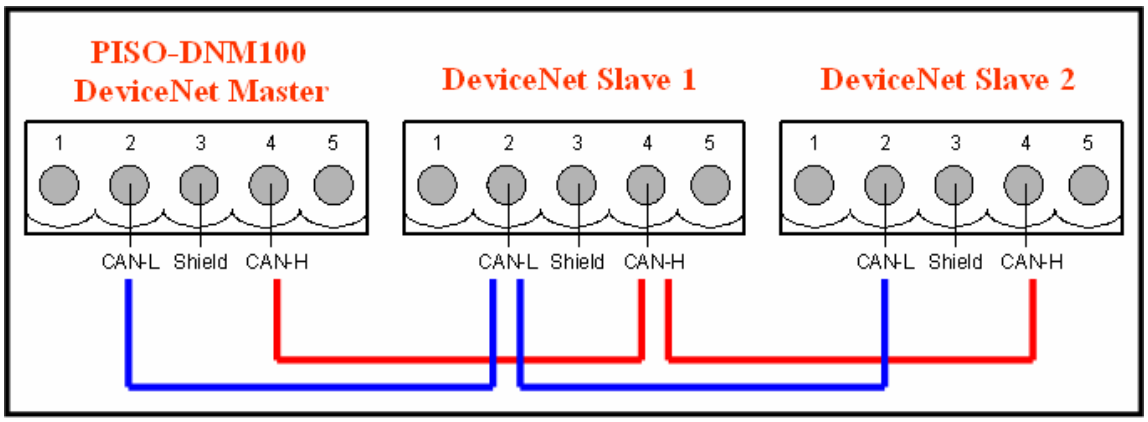


Figure 5.1 Correct wire connection

The following wire connection is **wrong** which is shown as Figure 5.2

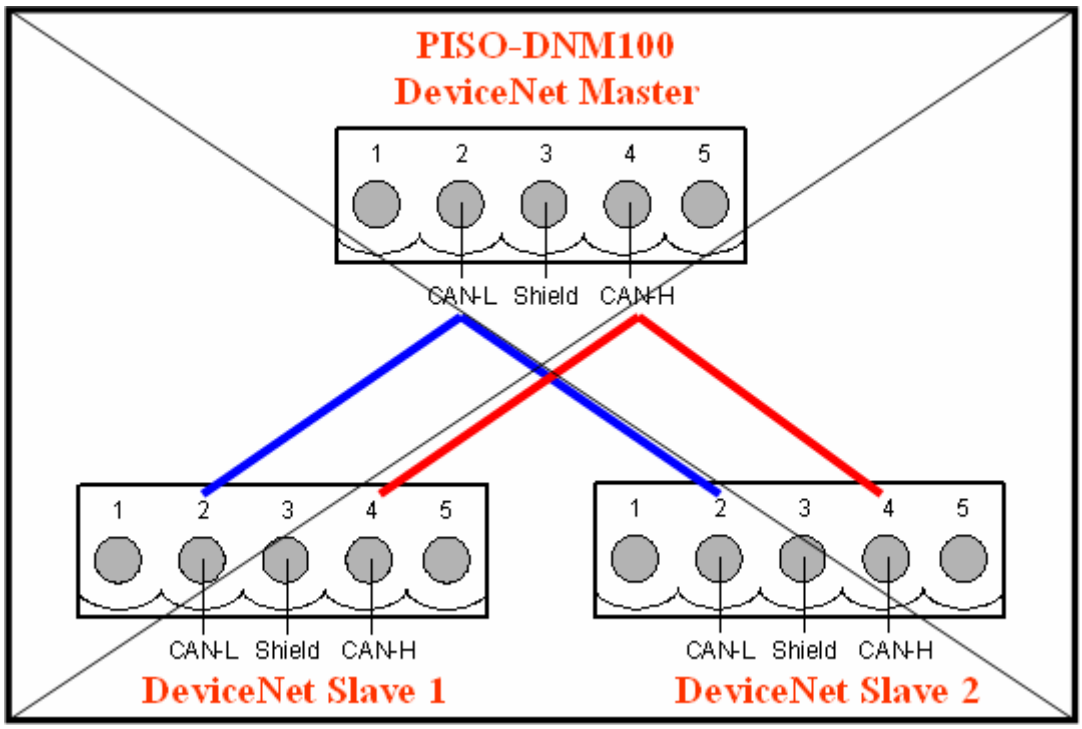


Figure 5.2 Wrong wire connection

5.3 VC_Demo1 Introduction

VC_Demo1 is the example used for starting the DeviceNet communication. The screen shoot is shown as Figure 5.3. This demo program is designed to communicate with slave device step by step. This program will read the input value of the slave device when the POLL connection has been established. Before exercising this demo, the users should have at least one DeviceNet slave device which has input channels (AI or DI) and finish the wire connection between the Master and slave device. (See Figure 5.1)

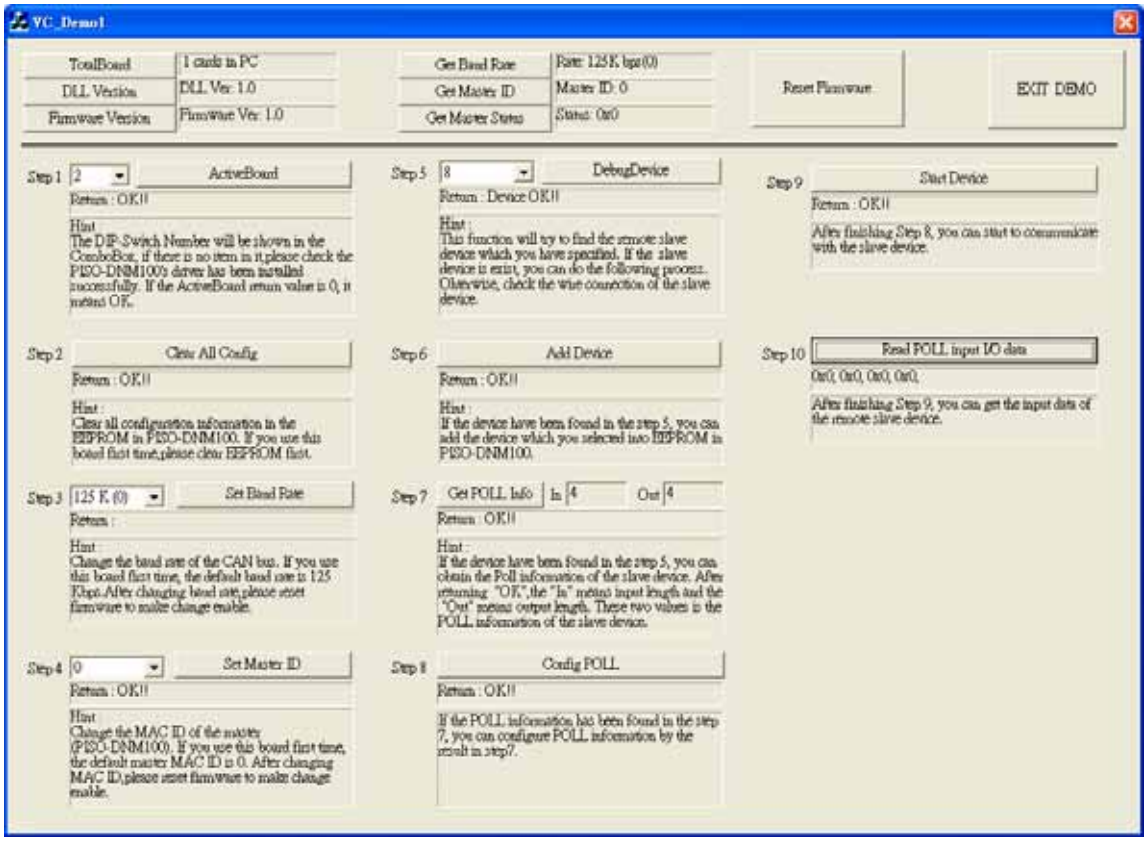

VC_Demo1
TotalBoard
1 cards in PC
DLL Version
DLL Ver: 1.0
Firmware Version
Firmware Ver: 1.0
Get Baud Rate
Rate: 125K bps(0)
Get Master ID
Master ID: 0
Get Master Status
Status: 0x0
Reset Firmware
EXIT DEMO
Step 1
ActiveBoard
Return : OK!!
Hint :
The DIP-Switch Number will be shown in the ComboBox, if there is no item in it,please check the PISO-DNM100's driver has been installed successfully. If the ActiveBoard return value is 0, it means OK.
Step 2
Clear All Config
Return : OK!!
Hint :
Clear all configuration information in the EEPROM in PISO-DNM100. If you use this board first time,please clear EEPROM first.
Step 3
125 K (0)
Set Baud Rate
Return :
Hint :
Change the baud rate of the CAN bus. If you use this board first time, the default baud rate is 125 Kbps.After changing baud rate,please reset firmware to make change enable.
Step 4
0
Set Master ID
Return : OK!!
Hint :
Change the MAC ID of the master (PISO-DNM100). If you use this board first time, the default master MAC ID is 0. After changing MAC ID,please reset firmware to make change enable.
Step 5
8
DebugDevice
Return : Device OK!!
Hint :
This function will try to find the remote slave device which you have specified. If the slave device is exist, you can do the following process. Otherwise, check the wire connection of the slave device.
Step 6
Add Device
Return : OK!!
Hint :
If the device have been found in the step 5, you can add the device which you selected into EEPROM in PISO-DNM100.
Step 7
Get POLL Info
In 4
Out 4
Return : OK!!
Hint :
If the device have been found in the step 5, you can obtain the Poll information of the slave device. After returning "OK","In" means input length and the "Out" means output length. These two values is the POLL information of the slave device.
Step 8
Config POLL
Return : OK!!
If the POLL information has been found in the step 7, you can configure POLL information by the result in step7.
Step 9
Start Device
Return : OK!!
After finishing Step 8, you can start to communicate with the slave device.
Step 10
Read POLL input I/O data
0x0, 0x0, 0x0, 0x0,
After finishing Step 9, you can get the input data of the remote slave device.

Figure 5.3 the screen shoot of VC_Demo1

After running the program, the uses will see the “TotalBoard” information on the left and up corner of the screen. This function determinates how many PISO-DNM100 in your PC automatically. If it doesn’t find any board, the users should check that the windows driver has been installed successfully. Otherwise, if it has found at lease one board, the users can continue exercising the demo program.

Step 1 : ActiveBoard

Before performing other buttons, the “ActiveBoard” button should be clicked firstly. The DIP-Switch on the board means the ID of this board. The users should make sure that every board’s ID in your PC is unique. The drop-down list will show the board’s ID which the users have selected. After clicking the button, the return code will be 0. Otherwise, please check the windows driver has been installed successfully.

Step 2 : Clear All Config

To avoid unknown configuration in the PISO-DNM100, the users can push this button to clear all configuration in the board.

Step 3 : Set Baud Rate

The default baud rate is 125Kbps. If the users want to change the value, you can select the correct value then push the button. After changing the baud rate, the uses should reset the firmware in PISO-DNM100 by pushing the “ResetFirmware” button. Wait for 1 or 2 seconds then go to the next step.

Step 4 : Set Master ID

The default Master’s MAC ID is 0. If the users want to change the value, you can select the correct value then click the button. After changing the Master’s ID, the uses should reset the firmware in PISO-DNM100 by clicking the “ResetFirmware” button. Wait for 1 or 2 seconds then go to the next step.

Step 5 : Debug Device

Before performing this function, the users should set the MAC ID of the slave device and turn on it. In the demo, the user can select the MAC ID of the slave device then click “Debug Device” button. This function will try to find the remote slave device which the user has appointed. Wait for 5 seconds, if the slave device exists in the network, the return value will be 0. Otherwise, the device doesn’t exist or has no response. The users can go to next step until the problem of the slave device has been solved.

Step 6 : Add Device

Before performing this function, the users should use step 5 to find an exist device. This function is to add the device’s information which have be found at step 5 into EEPROM in the PISO-DNM100. If it is successful, the return value will be 0.

Step 7 : Get POLL Info

This function is to obtain the device's POLL information. After the slave device responses the data, it will show in the "In" and "Out". "In" means the input length of the slave device. "Out" means the output length of the slave device. If the slave device responses successfully, the return value will be 0.

Step 8 : Config POLL

If the step 7 is successful, the users can perform "Config POLL" button. This function is to add the device's POLL information into EEPROM in PISO-DNM100. If it is successful, the return value will be 0.

Step 9 : Start Device

If the step 8 is successful, the users can perform "Start Device" button. This function will communicate with the slave device which the users have configured in the previous steps. If it is successful, the return value will be 0.

Step 10 : Read POLL Input I/O Data

If the master is communicating with the slave device successfully, the users can read the input I/O data from the slave device in this step. This function will obtain the input I/O data and show them in byte (8-bits). If the users have 8 digits input, the return value will show 1 byte. Similarly, if the users have 16 digits input, the return value will show 2 bytes. And so on.

5.4 VC_Demo2 Introduction

VC_Demo2 is the example used for scanning the DeviceNet slave devices in the network. The screen shoot is shown as Figure 5.4. This demo program is designed to operate the master step by step. This program will show the information of all the slave devices in the network. Before exercising this demo, the users should have at least one DeviceNet slave device and finish the wire connection between the Master and slave device. (See Figure 5.1)

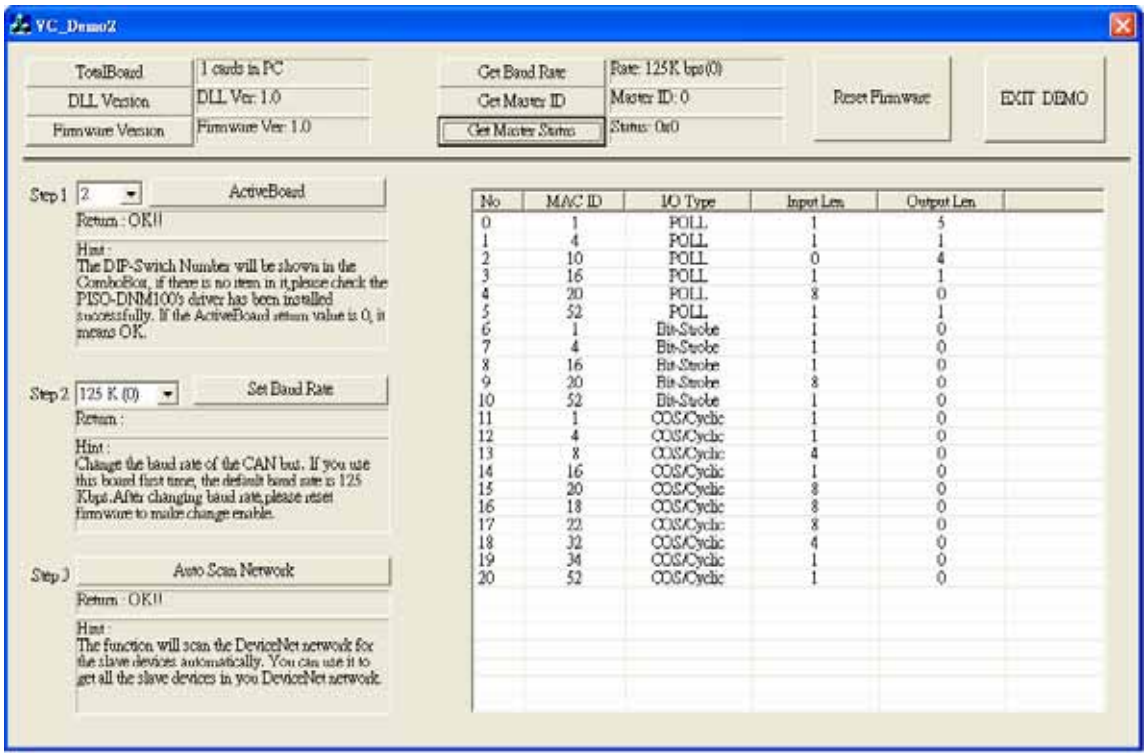

VC_Demo2
TotalBoard
1 cards in PC
DLL Version
DLL Ver: 1.0
Firmware Version
Firmware Ver: 1.0
Get Baud Rate
Rate: 125K bps(0)
Get Master ID
Master ID: 0
Get Master Status
Status: 0x0
Reset Firmware
EXIT DEMO
Step 1
2
ActiveBoard
Return : OK!!
Hint :
The DIP-Switch Number will be shown in the ComboBox, if there is no item in it,please check the PISO-DNM100's driver has been installed successfully. If the ActiveBoard return value is 0, it means OK.
Step 2
125 K (0)
Set Baud Rate
Return :
Hint :
Change the baud rate of the CAN bus. If you use this board first time, the default baud rate is 125 Kbps.After changing baud rate,please reset firmware to make change enable.
Step 3
Auto Scan Network
Return : OK!!
Hint :
The function will scan the DeviceNet network for the slave devices automatically. You can use it to get all the slave devices in you DeviceNet network.
No
MAC ID
I/O Type
Input Len
Output Len
0 1 POLL 1 5
1 4 POLL 1 1
2 10 POLL 0 4
3 16 POLL 1 1
4 20 POLL 8 0
5 52 POLL 1 1
6 1 Bit-Strobe 1 0
7 4 Bit-Strobe 1 0
8 16 Bit-Strobe 1 0
9 20 Bit-Strobe 8 0
10 52 Bit-Strobe 1 0
11 1 COS/Cyclic 1 0
12 4 COS/Cyclic 1 0
13 8 COS/Cyclic 4 0
14 16 COS/Cyclic 1 0
15 20 COS/Cyclic 8 0
16 18 COS/Cyclic 8 0
17 22 COS/Cyclic 8 0
18 32 COS/Cyclic 4 0
19 34 COS/Cyclic 1 0
20 52 COS/Cyclic 1 0

Figure 5.4 The screen shoot of VC_Demo2

After running the program, the uses will see the “TotalBoard” information on the left and up corner of the screen. This function determinates how many PISO-DNM100 in your PC automatically. If it doesn’t find any board, the users should check that the windows driver has been installed successfully. Otherwise, if it has found at lease one board, the users can continue exercising the demo program.

Step 1 : ActiveBoard

Before performing other buttons, the “ActiveBoard” button should be clicked firstly. The DIP-Switch on the board means the ID of this board. The users should make sure that every board’s ID in your PC is unique. The drop-down list will show the board’s ID which the users have selected. After clicking the button, the return code will be 0. Otherwise, please check the windows driver has been installed successfully.

Step 2 : Set Baud Rate

The default baud rate is 125Kbps. If the users want to change the value, you can select the correct value then click the button. After changing the baud rate, the uses should reset the firmware in PISO-DNM100 by clicking the “ResetFirmware” button. Wait for 1 or 2 seconds then go to the next step.

Step 3 : Auto Scan Network

Before performing this function, the users should set the MAC ID and the baud rate of the slave device and turn on it. In the demo, the user can click “Auto Scan Network” button to obtain all the I/O information of all slave devices in the network. Wait for 30 seconds. If the remote slave devices exist or have response, the users will see the entire slave device and their I/O information in the list table.

5.5 BCB_Demo1 Introduction

BCB_Demo1 is the example used for scanning the DeviceNet slave devices in the network. The screen shoot is shown as Figure 5.5. This program will show the information of all the slave devices. This demo is similar to VC_Demo2. Before exercising this demo, the users should have at least one DeviceNet slave device and finish the wire connection between the Master and slave device. (See Figure 5.1) The users can configure the slave device by the scanning information. This demo can be a tool to add or remove the configuration of the slave device.

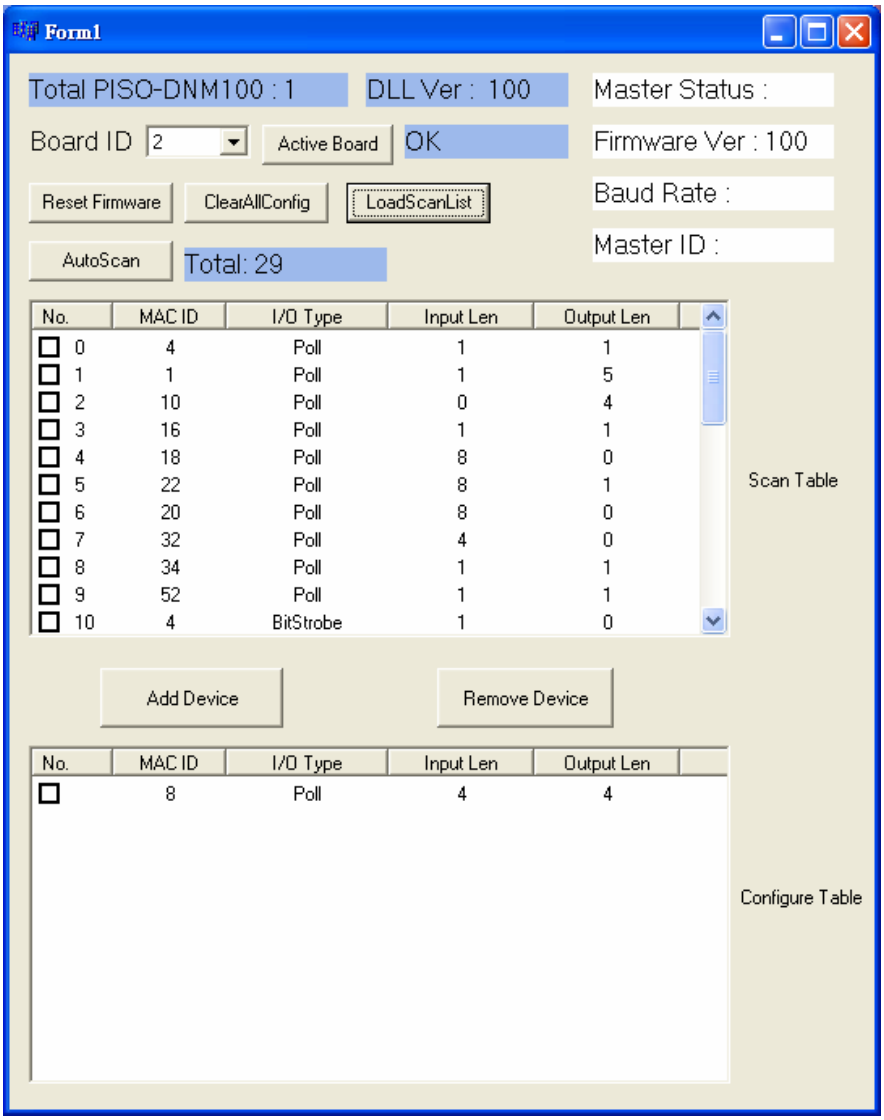


Figure 5.5 The screen shoot of BCB_Demo1

After running the program, the users will see the “Total PISO-DNM100 : x” information on the left and up corner of the screen. This function determinates how many PISO-DNM100 in your PC automatically. If it doesn’t find any board, the users should check that the windows driver has been installed successfully. Otherwise, if it has found at lease one board, the users can continue exercising the demo program.

Step 1 : Active Board

Before performing other button, the “Active Board” button should be clicked firstly. The DIP-Switch on the board means the ID of this board. The users should make sure that every board’s ID in your PC is unique. The drop-down list will show the board’s ID which the users have selected. After clicking the button, the return code will be 0. Otherwise, please check the windows driver has been installed successfully.

Step 2 : Auto Scan

Before performing this function, the users should set the MAC ID and the baud rate of the slave device and turn on it. In the demo, the user can click “Auto Scan” button to obtain all the I/O information of all slave devices in the network. Wait for 30 seconds. If the remote slave devices exist or have response, the users will see the entire slave device and their I/O information in the “Scan Table”.

Step 3 : Add Device

This function is to add the device’s information into EEPROM in the PISO-DNM100. The users can check the item which you want to add. After checking what you want, push “Add Device” button to add the information into the EEPROM. If it is successful, the items which you selected will be shown in the “Configure Table”.

Step 4 : LoadScanList

This function is to obtain the device’s information from EEPROM in the PISO-DNM100. The users can check the information in the EEPROM. After performing the function, the information will be shown in the “Configure Table”.

5.6 BCB_Demo2 Introduction

BCB_Demo2 is the extension of the BCB_Demo1. The screen shoot is shown as Figure 5.6. This program can read the input data and write the output data every one second. This demo is similar to BCB_Demo1. We just introduce the extension part. Before exercising this demo, the users should have at least one DeviceNet slave device and finish the wire connection between the Master and slave device. (See Figure 5.1) The users can configure the slave device by the scanning information. This demo can be a tool to add or remove the configuration of the slave device. Additionally, the users can operate the I/O data form the remote slave devices.

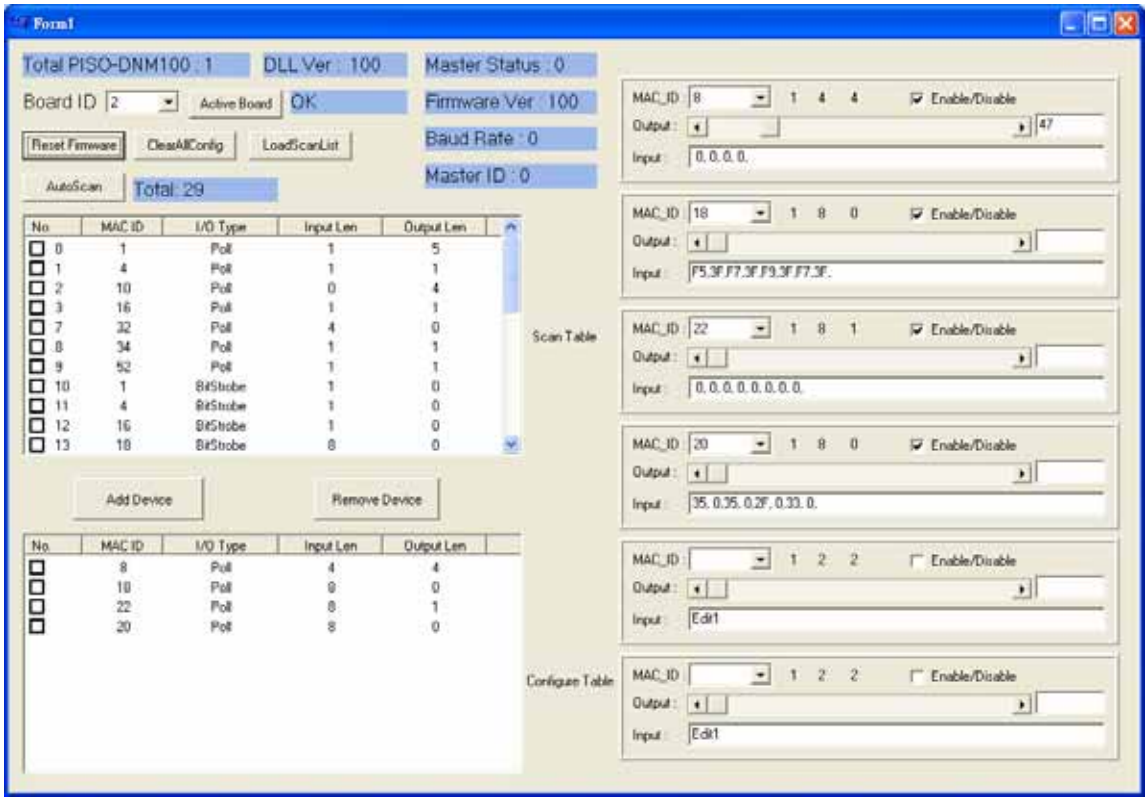


Figure 5.6 The screen shoot of BCB_Demo2

If the users want to know how to configure the slave device information into EEPROM in the PISO-DNM100, please refer to section 5.5.

If “Active Board” is OK, the users can click “LoadScanList” button. The configuration information will be shown in “Configure Table”. At the same time, the MAC IDs also are shown on the right side of the screen. The users can check the “Enable/Disable” box to enable or disable the read and write the I/O data. The “Output” scroll bar presents the output value which will be written to the output channel of the slave device. If the scroll value is 0x23, every byte of the slave device’s output is 0x23. The users can find out the change of the output channel easily. The “Input” field presents the input value which has been read from the input channel of the slave device.